

PAPER • OPEN ACCESS

A unique message encryption technique based on enhanced blowfish algorithm

To cite this article: Godfrey L Dulla *et al* 2019 *IOP Conf. Ser.: Mater. Sci. Eng.* **482** 012001

View the [article online](#) for updates and enhancements.



IOP | ebooks™

Bringing you innovative digital publishing with leading voices to create your essential collection of books in STEM research.

Start exploring the **collection** - download the first chapter of every title for free.

A unique message encryption technique based on enhanced blowfish algorithm

Godfrey L Dulla¹, Bobby D Gerardo², and Ruji P Medina³

Department of Graduate School, Technological Institute of the Philippines, Quezon City, Philippines

gldulla@gmail.com¹, bgerardo@wvsu.edu.ph², and ruji_p_molina@gmail.com³

Abstract. Online transaction especially sending of messages has become easier and simpler to be manipulated. Nonetheless, security is the important aspect to secure its content and raise difficulty to unauthorized attempt. This study aims to secure both plaintext and file message content through encryption technique which is based on enhanced Blowfish algorithm. An enhanced Blowfish algorithm is developed to improve its performance by reducing the number of rounds and by increasing the block length with fixed length during encryption and decryption with added transformation method on selected rounds. The study used Visual Studio application running at x64 operating system.

1. Introduction

There are various techniques and methods designed for specific problem. Encryption of data over the network is one of the key method which translates information that only authorized user has the knowledge of the secret key. The purpose of encryption is to protect and secure confidential information transmitted over communication channel/medium. Not all encryption initiatives gained acceptance and popularity because of different reasons, Blowfish is not an exemption. Blowfish algorithm used 64-bit block cipher which is said to be vulnerable to attacks as opposed to newer algorithm that uses larger block sizes. Many cryptographers examined blowfish but few have published. Some have examined and detected weak keys on certain rounds and adjustments on variable sizes but others have successfully implemented in software and hardware with less memory consumption and improved execution time. Nonetheless, Blowfish potentially offers significant exploration consideration given that various researches suggest to further work on variable size, expansion of key length and iterations.

Thus, this study aims to explore the processes involved by reducing the number of rounds and increasing the block variable length and introducing transformation method on selected rounds as attempt to introduce new method in encrypting the message and compare the results with the existing algorithm. The enhancements to the cryptanalysis provide better security for the message and amplify potential advantages to Blowfish algorithm.

2. Review of Related Literature

There is a general need for enciphering all data sent between computers connected to a network. The communication lines between different terminals are wide open not only to tapping but also to deliberate or accidental alteration and corruption of traffic [1]. Computer systems must have a means to prevent



unauthorized access to avoid any alteration or worse loss of data between devices connected to the network.

The Blowfish Encryption Algorithm contains 16 rounds; each round consists of XOR operation and a function (F). Blowfish is fast, compact, simple and variably secure. Significantly faster than DES and optimized for application where key does not change like communication link or file encryptor and not suitable for packet switching with frequent key changes or one-way hash function. A feistel network is consist of two parts – key expansion and data encryption. The key expansion converts a key of up to 448 bits into several subkey arrays totaling to 4,098 bytes. Data encryption consists of a simple function iterated 16 times. Each round consists of permutation and key. And, all operations are additions and XORs on 32bit words [2].

A combined hybrid cryptosystem along with splitting and merging mechanism enhanced the speed, time and security of the file using SRNN cipher algorithm. It includes file splitting and merging mechanism where each slice is encrypted by its corresponding key [3][4] developed a tool designed for file encryption. That is, the file was divided into number of pieces which depends on user's specification with application of encryption algorithm. They modified the F-function defined as $F(X) = ((S1 + S2 \bmod 232) \text{ XOR } S3 + S4 \bmod 232)$ into $F(X) = ((S1 \text{ XOR } S2 \bmod 232) + (S3 \text{ XOR } S4 \bmod 232))$. As a result, they proved that the total execution time of the modified algorithm is 14% lesser than the original Blowfish algorithm. [5] presented a method for image encryption/decryption designed to increase the security and improve the performance using a variable key size of 448 bits employing 16 iterations. The data image served as the plaintext with two input of encryption process. The image data bit stream is divided into the block length and excludes the image header. The start of the bitmap pixel or array begins right after the header of the file. The byte elements of the array are stored in row order from left to right with each row representing one scan line of the image and the rows of the image are encrypted from top to bottom. They concluded that both colored and white image of any size saved in tagged image file format (TIF), bitmap (BMP), portable network graphics (PNG), joint photographic experts group (JPG), etc can be successfully encrypted and decrypted. The algorithm cannot be broken until an attacker attempted 2^{8r+1} combination (r is the number of rounds). Furthermore, they concluded that the algorithm becomes stronger if the number of round are increased thereby considered as an excellent standard encryption algorithm. Ghorpade and Talwar [6] and Sowbarnika, Balasubramani, and Varatharajan [7] pointed out that text and pictures/video files have different prerequisite in the context of data encryption. Usually text is smaller than image and when both use the same algorithm to encrypt and decrypt, the data image will more likely perform slower than text. This is true with data on the internet where images need to be protected more than text. The Blowfish was utilized by the authors as they consider it as the best encryption algorithm with a good performance and only consume as low as 5K of memory. The authors have compared this algorithm with other symmetric encryption algorithms like AES and DES but to them Blowfish are capable of variable length key which makes it more difficult to decode the key from attackers. Enhanced-Blowfish is in par with Blowfish algorithm only that it is more secured due to block switching as an additional means of scrambling data especially information must be the topmost security concern in today's business and industry.

Agrawal and Mishra [8] enhanced the security level of Blowfish algorithm and decrease the time for encryption and decryption. They generate random number from 0 to 65535 and number into 16 bit binary form and find the positions that are holding 0 entries. If the flag is 1 then the F-function will not work, if the flag is 0 then the F function will work. In every round a new random number is generated and this gives different results in the application of F function. They analysed that the encryption and decryption time and concluded that it was reduced compared to the original Blowfish algorithm. Similar study was conducted [9], the generated random numbers come from an image. It reads the picture by pixel, selects the specific location randomly picked at any two colors then applied XOR between two selected colors then specify the length of the key. On the other hand, dividing the original image into a random number of blocks and then transformed the blocks into new locations give better transformation because aside from fewer pixel it kept their neighbors and difficult to predict neighbor pixels. The entropy was increased providing better security for the image [10].

Manku and Vasanth [11] developed a system using blowfish algorithm that reduces the rounds and made modification in each round. They proposed single blowfish round with two S-boxes connected with XOR same as the other 2 S-boxes. The 2 XOR was added then form the plaintext. The system was simulated and implemented in VHDL and Java jdk1.7 using Eclipse 4.2.0 integrated environment. In the study of [12] the implementation of the Function F was changed because the change in the total time taken for encryption and decryption cannot be seen on software implementation. The authors implemented VHDL application to show the differences in the delay. [13] They concluded that the improved modified algorithm has enhanced the performance over existing blowfish algorithm by reducing the number of clock cycles required for the execution of Blowfish function by 33% and hence reducing the overall execution time of the modified Blowfish algorithm by 14%. A sample waveforms were used because the modified Blowfish function F' executes both the summations in parallel, whereas the existing function executes the sums in sequential fashion. [14] in their comparative analysis that increase in processing rounds strengthens the security as single Fiestel round provides inadequate security.

Panda [15] studied to evaluate the amount of computing resources consumed namely, CPU time, memory, and battery power using AES, DES, Blowfish and RSA algorithms. The researcher used three different types of files (text, binary and image) using encryption time, decryption time and throughput as performance parameters. The AES provided the better performance in terms of both throughput and encryption-decryption.

Ramesh and Suluriandi [16] yielded a different result. The algorithms tested were DES, AES and Blowfish against certain performance metrics namely, execution time, required memory for implementation and throughput. The experiments resulted in concluding that Blowfish is the best performing algorithm among the tested algorithms. The paper further supports this by stating that Blowfish performs approximately 4 times faster than AES and 2 times faster than DES while consuming less memory in the process.

There have been studies conducted on comparison of the different algorithms such as DES, 3DES, AES, Blowfish, RC4, MD5, IDEA, TEA, CAST, RC2, RC5, RC6, Serpent, Twofish, Threefish, Mars, ECC, DHA and SHA. The blowfish and ECC provided the faster encryption speed. ECC is having some attacks on it but on Blowfish no attack was successful [17]. And, blowfish is much faster than DES but the speed increasing for Blowfish is slower compared to DES because it needs much more memory for sub-key and S-box initialization [18]. But, there is a minimal performance and low overhead cost achieved by Elliptic Curve Digital Signature Algorithm (ECDSA) while still providing protection for the outsourced data [19].

Priya and Arun [20] showed that Blowfish is most efficient and consumes less memory. This is a surprising and interesting result since AES supersedes Blowfish in speed but considering the efficiency of Blowfish algorithm where even AES could not count on. It cannot be taken for granted especially the need to conserve power consumption in mobile and ubiquitous computing is something Blowfish algorithm is worth considering as a valuable option. Enhanced-Blowfish wraps this up for an added security as the core means of enhancing the stability of Blowfish algorithm.

Ahmad, Manaf and Ismail [21] found that Blowfish has minimum power consumption (microJoule/Byte) and percent battery consumed in his evaluation of performance of the symmetric key algorithms: DES, 3DES, AES and Blowfish. Also, [22] found blowfish proved to be superior for secure wireless communication with high speed using low power source of energy efficiency. They concluded that the algorithm is best embedded in mobile devices and it was verified through a reprogrammable FPGA. The optimal performance is defined strictly by the least number of hardware requirements, highest throughput, and lowest power consumption. The results showed the algorithm has the smallest design core and highest throughput, with low power consumption. The same with the study made by Mandal, it was found that Blowfish is the most efficient one in terms of power consumption, security and throughput.

3. The Enhanced Encryption Technique

In this study, the concept follows the reverse, swapping and shifting of plaintext to its binary equivalent. A series of steps have been implemented to produce the encrypted values. The 32-bits representation in each block carries the plaintext in byte before converting it to its equivalent ASCII representation. The final result shall be transformed into its equivalent encrypted values.

3.1. Design Concept

In Figure 1, the uploaded file and inputted key supplied by the user served as baseline parameters for the entire process. Each phase has its own set of scheme allowing the result to be a strong potential ciphertext of that given process. Reducing the number of rounds and increasing the block length to four (4) blocks with 32-bit fixed length during encryption and decryption with added transformation method on selected rounds are main components to transform the message to generate the secured ciphertext.

3.2. Encryption/Decryption

In the encryption and decryption process, the series of steps follows the shifting, XOR and switching scheme. The block length will be increased to 128-bit and the number of rounds will be reduced from 16 to 14 rounds. The key will be provided by the user and generation of random characters and numbers that will form the plaintext. The integration of ShiftLeft and ShiftRight (according to the file size) and swapping method in each round posed potential adjustment on the speed, efficiency and generation of strong ciphertext.

In Figure 1, the plaintext will be divided into 4 blocks with 32-bit fixed length size. Each block has each own structure and operation. In the first 10 rounds, the operation will consist of XOR, Shifting and swapping scheme. The remaining rounds include the swapping of the center blocks with combination XOR and another swapping operation.

Many algorithms like DES, AES, XXTEA, MARS and many others make use of shift as part of the cycle (rounds). The shift method was used in the new version of blowfish called Twofish using 128-bits. The shift was introduced in K1 to K3 using $\lll 8$, $\lll 1$ and $\ggg 1$ respectively. In the proposed study, the shift method will be used in each block directed to the function with the S-box. Figures 4 and 5 shows the proposed enhancement to the algorithm. This gives a conditional process that will depend on the file input. There will be two sets of operation involved in the process. If the file size of the input resulted as even value the shifting will use the ShiftLeft method. This scheme will provide additional security as it will be difficult to predict the file size.

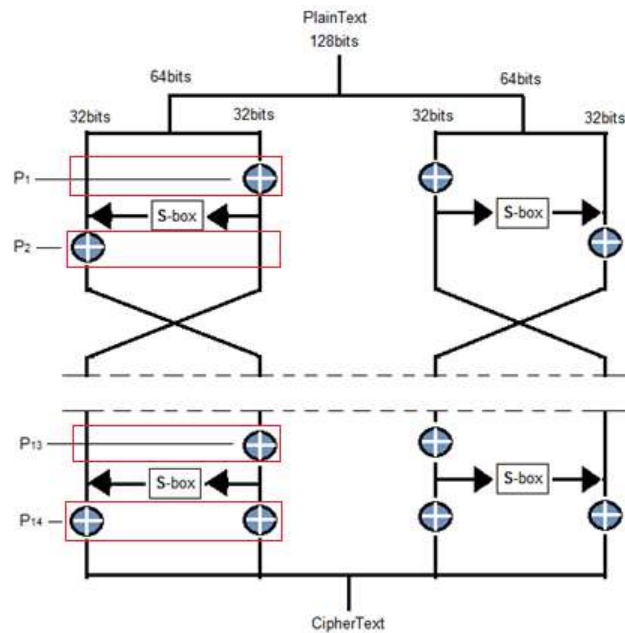


Figure 1. Enhancement Concept for Encryption and Decryption

The enhancement in the S-box perceived to be interesting that requires activity as compared with the iterations

$$F = ((S4 \text{ XOR } S3, \text{Shift2}(a) + S2), b \text{ XOR } \text{Shift4}(S1)) \quad (1)$$

The input will be divided into four (4) 32bits block and each block consists of 8bits. From the S-boxes the input will be transformed into 32bits following the lookup table that consists of 256 hexadecimal values (00 to FF) and the output will be 32bits. One thing that this proposed study is interesting (as mentioned earlier) is the double layered process (from 32bit to 8bit operation). The process embedded in the 32bits process and 8bits process presents the inner and outer process with parallel operation.

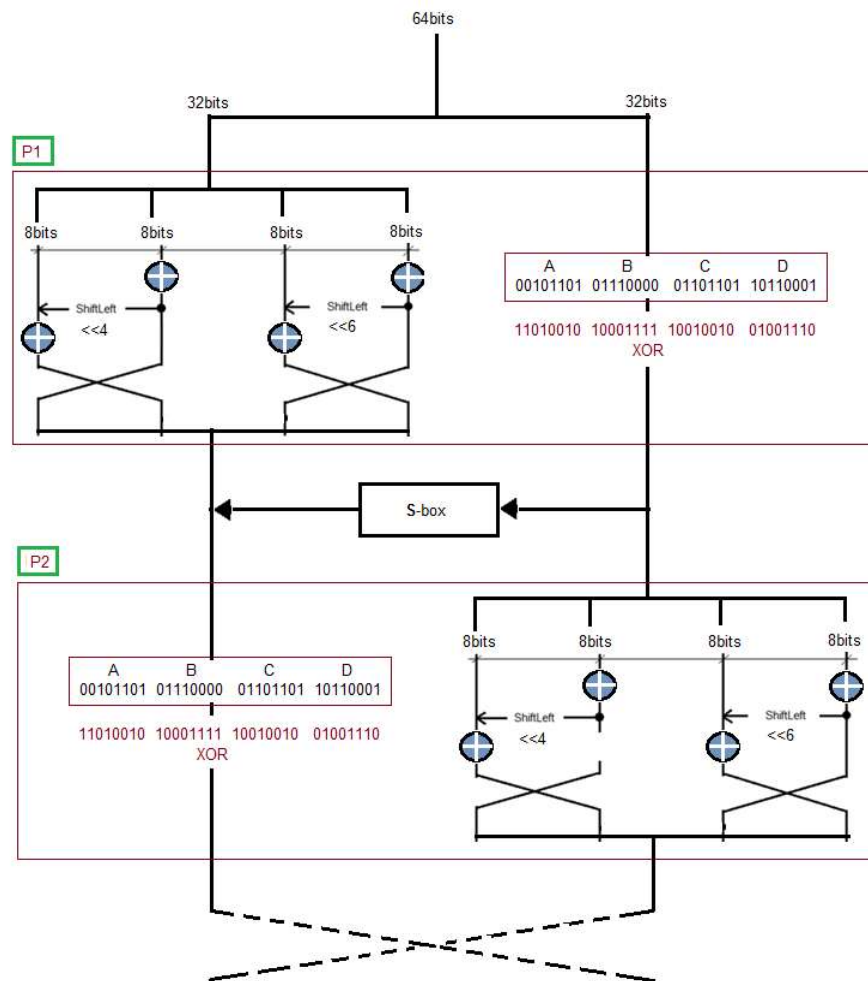


Figure 2. Detailed Proposed Enhancement Showing the Coverage of Each P-Array

Figure 2 illustrates the detailed proposed enhancement showing the coverage of each P-array. Each P-array consists of XOR and shifting of bits in the 8-bits phase which completes the cycle. After, the S-box will be performed, preparing each 32bits rounds for another P-array procedure.

4. Results and Discussions

The encryption/decryption process provides the file content to be transformed in such a way that only the receiver enables to reveal its content. The parameter filesize and generated key (from combination string of characters, numbers, and location) offer to enhance the algorithm leading to the generation of ciphertext. Figure 3 shows the main user interface for encryption and decryption process.

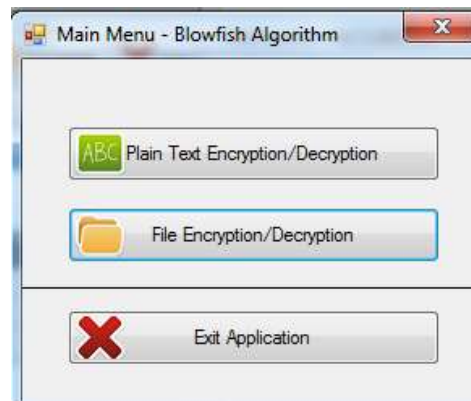


Figure 3. Main interface of the Program

The user submits the file for encryption and generates the ciphertext. The proposed algorithm was simulated in two options: plaintext and file content. In plaintext message, the user needs to input the text to be processed. The input contains any symbols or characters and holds closed to 33,000 characters (Figure 4).

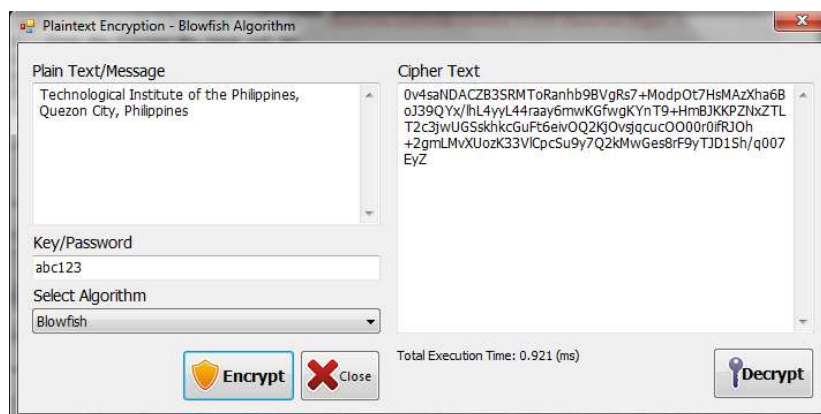


Figure 4. Encryption interface showing an encrypted message

The characters with its equivalent encrypted values formatted in hexadecimal format (00 to FF) follows the proposed enhancement algorithm as provided in the proposed section. A key or password provided by the user served as added information to the encryption process allowing the user to redisplay the original message. As an example, the user inputted the key as abc123 and the message translated into its corresponding equivalent. In the decryption process, in the event that user failed to submit the correct key or password, the message will not be reverted back to its original message but rather different ciphertext message will be generated (Figure 5).

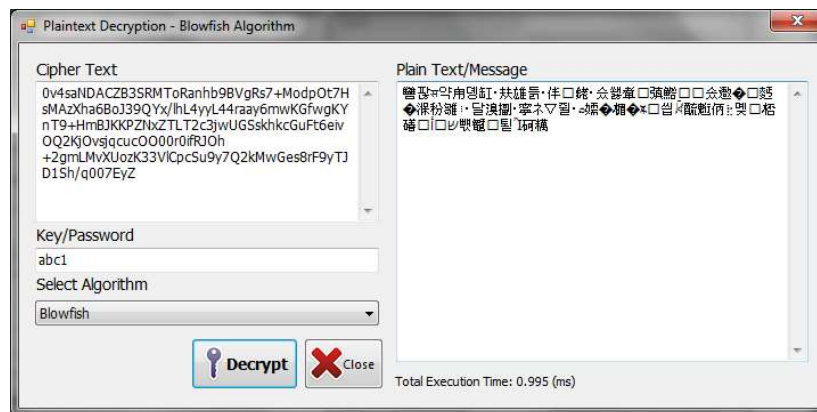


Figure 5. File encryption interface showing the encrypted file content

The inputted key or password (reflected as abc1) which is incorrectly typed generated different results. In the case of file content encryption, the user needs to select and open the file for processing. In the example, the user opened the file named sample.txt together with the key or password ABCDEFGH and displayed in the file content window, the cipher text window generates the encrypted text of the file (Figure 6). The encrypted file can be saved in .txt file format. To decrypt the file content, the user needs to upload the encrypted file to allow the decryption process decrypt said file content (Figure 7).

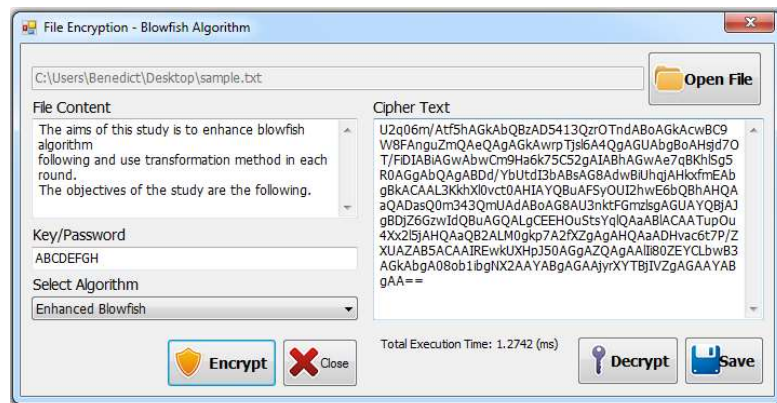


Figure 6. File encryption interface showing the encrypted file content

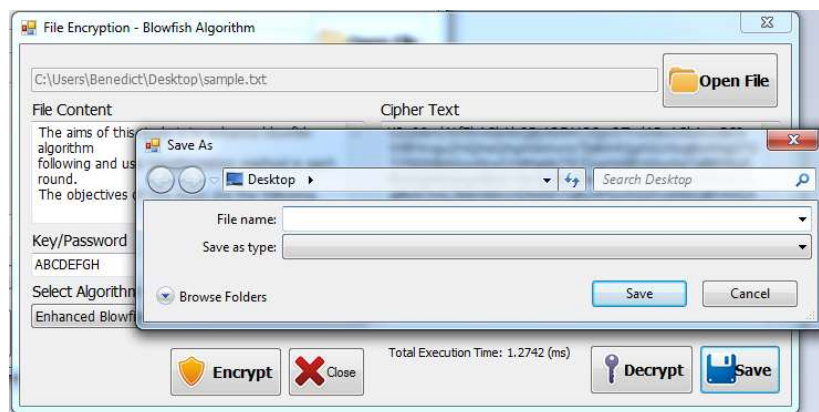


Figure 7. Saving ciphered text to assigned file name

The succeeding section presents the simulation results of the algorithm both for plain text and file content showing a significant change in the execution time following the modification in the iterations and block length expansion. Table 1 shows the comparative runtime performance of encryption and decryption process. The program used different filesize in text file format. The improved modified algorithm has enhanced the performance by reducing the number of rounds, expanded block size along with the different activities of bit-byte manipulation, shifting and block swapping by 11.37% or 2.69 msec advantage over the classic algorithm.

Table 1. Encryption comparative runtime performance (in milliseconds)

Key	filesize (kb)	Classic (ms)	Enhanced Blowfish (eBf)	Difference
file1	64682	32.64	5.34	16.37%
file2	17951	13.40	1.74	13.02%
file3	35902	22.99	2.06	8.95%
file4	29960	21.40	2.22	10.35%
file5	37840	28.08	2.11	7.51%
	AVG	23.70	2.69	11.37%

This shows that the modification made in the iterations affects the execution time without compromising the intricacy of the results (as previously presented). The file with smaller file size (file2) having 17951kb was executed in 13.40ms using the classic Blowfish algorithm whereas the enhanced algorithm has a result of 1.74ms or a difference of 13.02%. The file1 has the highest file size (64682kb), in classic algorithm it was executed in 32.64ms, and 5.34ms for the enhanced algorithm. The table presents that the enhancements made show significant change in execution time.

In decryption performance, the process was improved by 10% or 2.29msec of the average runtime performance over the classic algorithm. The algorithm depends on the filesize, that is, it delivers an improved performance in terms of smaller file size. For example, the file size with 172505 (being the higher file size) performed 9% over 34.16 (classic) and 3.12 (enhanced) decryption processing time. This provides the same performance result that was observed when a file has a file size of 100k-175k. This was also observed in file size 95k-80k which performed 10% over the existing algorithm.

Table 2. Decryption comparative runtime performance of classic and Enhanced Algorithm (in milliseconds)

Key	filesize after encryption (kb)	Classic (ms)	Enhanced Blowfish (eBf) (ms)	Difference
efile1	172505	34.16	3.12	9%
efile2	47873	13.87	1.72	12%
efile3	95745	21.60	2.27	10%
efile4	79897	21.78	2.08	10%
efile5	100909	25.56	2.28	9%
	AVG	23.39	2.29	10%

The results showed that the enhanced encryption to the algorithm has provided the fastest execution time compared with the classic encryption algorithm (Figure 8). The reduced number of iterations to the enhanced algorithm affects this performance.

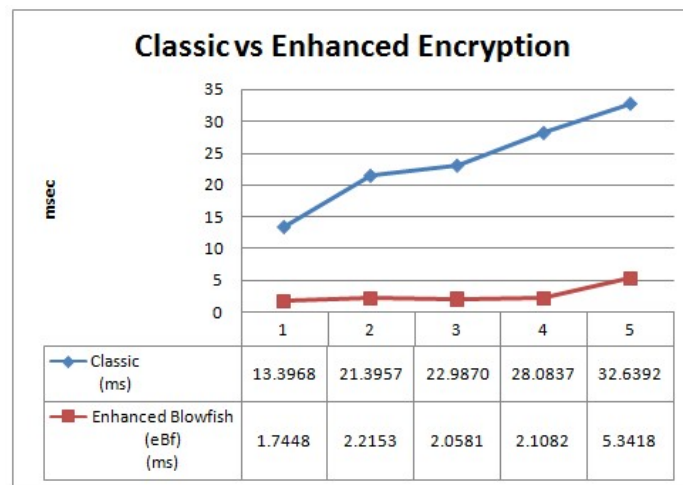


Figure 8. Simulation Results of Classic vs Enhanced Encryption

In decryption results, it showed that same with the encryption process, the enhanced algorithm provided the fastest execution time in dealing with decrypting file content. In the enhanced algorithm, the filesize plays significant factor in translating the file content. The higher the value of the decrypted file produced, the longer processing time. There was a small difference in processing the file with size 47k-79k which is 1.72ms and 2.08ms having a difference of only .36ms (Figure 9).

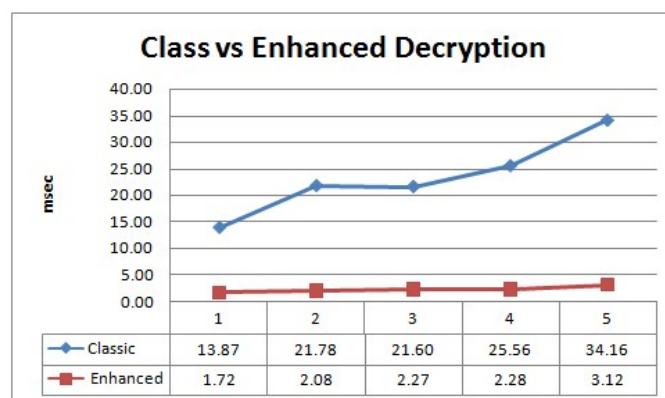


Figure 9. Simulation Results of Classic vs Enhanced Decryption

5. Conclusion

The enhancements made in the Blowfish Algorithm have provided a significant changed in the execution time without compromising the complexity of the encrypted file content. The reduce in the number of iteration gives the algorithm a faster execution time. And, the enhancements in the block size and other manipulation such as application of byte splitting, block size transposition, shift row and mix row methods provided complexity to the encrypted file content. Thus, it offers a considerable encryption performance of 2.69ms or 11.35% advantage over the existing Blowfish algorithm. And, a considerable decryption performance of 2.29ms or 9.8% difference.

The researchers concluded that this paper presents an enhanced Blowfish cryptanalysis that will improve its performance without compromising the complexity of the entropy which can be used in securing data transmitted over an unsecured communication channel.

Acknowledgment

The author/s would like to extend their sincere thank you first to Almighty God for all the blessings. Secondly to my family, friends, and relatives who extended their full support in the fulfillment of his Doctor in Information Technology and this paper. Lastly, to the three Institutions, St. Joseph's College of Rodriguez, Technological Institute of the Philippines and Commission on Higher Education Department (CHED) for all the push and scholarship grant that made this a reality.

References

- [1] Feistel H 1973 Cryptography and computer privacy *Scientific American* vol **228** no **5** pp 15-23
- [2] Schneier B 1996 *Applied Cryptography, Second Edition: Protocols, Algorithms, and Source Code* (John Wiley & Sons)
- [3] Sharma M and Sharma V 2016 A hybrid cryptosystem approach for file security by using merging mechanism *2nd Int. Conf. on Applied and Theoretical Computing and Communication Technol. (ICATCCT)* (IEEE) 978-1-5090-2399-8 pp 713-7
- [4] Manikandan G, Rajendran P, Chakarapani K, Krishnan G and Sundarganesh G 2012 A modified cryptoscheme for enhancing data security *J. of Theoretical and Applied Inform. Technol.* vol **35** no **2** pp 149-54
- [5] Singh P and Singh K 2013 Image encryption and decryption using blowfish algorithm in Matlab *Int. J. of Sci. & Eng. Res.* vol **4** no **7** retrieved from website: <https://www.ijser.org>
- [6] Ghorpade A and Talwar H 2016 *The Blowfish Algorithm Simplified* ISSN: 2278-8875
- [7] Sowbarnika A, Balasubramani M and Varatharajan N 2017 Evaluating the effects of cryptographic algorithms on different sets of text data *Int. J. of Comp. Sci. and Eng.* (IEEE) pp 152-60
- [8] Agrawal M and Mishra P 2012 A modified approach for symmetric key cryptography based on blowfish algorithm *Int. J. of Eng. and Adv. Technol.* vol **1** no **6** pp 79-83
- [9] Tahseen I and Habeeb S 2012 Proposal new approach for blowfish algorithm by using random key generator *J. of Madent Alelem College* vol **4** no **1** pp 1-10 retrieved from website: <https://www.iasj.net>
- [10] Devi A, Sharma A and Rangra A 2015 A review on DES, AES and Blowfish for image encryption and decryption *Int. J. of Comp. Sci. and Inform. Technol.* vol **6** no **3** pp 3034-6 ISSN:0975-9646
- [11] Manku S and Vasanth K 2015 Blowfish encryption algorithm for information security *ARPJ. of Eng. and Applied Sci.* vol **10** no **10**
- [12] Krishnamurthy G N, Ramaswamy D V and Leela M G 2007 Performance enhancement of blowfish algorithm by modifying its function *Innovative Algorithms and Techniques in Automation, Industrial Electronics and Telecommunications* pp 241-4
- [13] Kondawar S and Gawali D 2016 Security algorithms for wireless medical data *Int. Conf. on Green Engineering and Technologies (IC-GET)* (IEEE)
- [14] Zaran H, Druga G and Varazdin C 2016 Comparative analysis of cryptographic algorithms *Int. J. of Digital Techn. and Economy* vol **1** no **2** pp 127-34
- [15] Panda M 2016 Performance analysis of encryption algorithms for security *Int. Conf. on Signal Processing, Communication, Power and Embedded System (SCOPEs) 2016* pp 278-84
- [16] Ramesh A and Suruliandi A 2013 Performance analysis of encryption algorithms for information security *Int. Conf. on Circuits, Power and Computing Technologies (ICCPCT-2013)* pp 840-4
- [17] Bhanot R and Hans R 2015 A review and comparative analysis of various encryption algorithms *Int. J. of Security and Its Applications* vol **9** no **4** pp 289-306 <http://dx.doi.org/10.14257/ijisia.2015.9.4.27>
- [18] Nie T and Zhang T 2009 A study of DES and blowfish encryption algorithm *A Project of Shandong Province Higher Educational Science and Technology Program* (No. J09LG10) (IEEE 978-1-4244-4547-9)

- [19] Gajra N, Khan S and Rane P 2014 Private cloud security: Secured user authentication by using enhanced hybrid algorithm 2014 *Int. Conf. on Advances in Communication and Computing Technologies*
- [20] Priya S and Arun B 2017 Implementation of blow fish algorithm in crypto system for text and file data hiding *Int. J. of Sci. Res. Organization* vol **1** no **3** pp 29-35 e-ISSN: 2456 6942
- [21] Mandal P C 2012 Superiority of blowfish algorithm *Int. J. of Adv. Res. in Comp. Sci. and Soft. Eng.* vol **2** no **9** pp 196-201
- [22] Ahmad R, Manaf A A and Ismail W 2016 Implementation of a high-performance blowfish for secure wireless communication *J. of Telecom., Electronic and Comp. Eng.* vol **8** no **6** pp 147-51 ISSN: 2180-1843, e-ISSN: 2289-8131 retrieved from <http://journal.utem.edu.my/index.php/jtec/article/view/1264/755>
- [23] Mandal P C 2012 Evaluation of performance of the symmetric key algorithms: DES, 3DES, AES and Blowfish *J. of Global Res. in Comp. Sci.* vol **3** no **8** pp 67-70