ISBN 978-981-11-7861-0
Proceedings of 2018 the 8th International Workshop on Computer Science and Engineering (WCSE 2018)
Bangkok, 28-30 June, 2018, pp. 578-584
doi: 10.18178/wcse.2018.06.097

Blowfish-128: A Modified Blowfish Algorithm That Supports 128-bit Block Size

Ariel Roy L. Reyes [+], Enrique D. Festijo and Ruji P. Medina

Graduate Programs, Technological Institute of the Philippines, Quezon City, Philippines

Abstract. Symmetric crypto-algorithms are fast and extremely secure through strong key size. Among symmetric crypto-algorithms, Blowfish has shown superiority in performance amongst others and demonstrated excellent security as it is known to have no successful attacks. However, Blowfish algorithm became unfavorable to be used because of little block size support despite being known as an unbroken algorithm. In the present study, a novel modified version of the Blowfish cryptographic algorithm designed to support 128-bits block size input using dynamic selection encryption method and reduction of cipher function execution through randomly determined rounds is introduced. Experimental results on varied input lengths revealed that the modifications improved on performance and execution time, and increased the degree of complexity, and diffusion. The modification provided an additional layer of security that makes the virtual world safer and more resilient to attacks by unauthorized parties.

Keywords: avalanche, blowfish, cryptography, dynamic selection method, throughput.

1. Introduction

Security issues on personal information are considered as one of the most important challenges and biggest concern in today's connected world [1], [2]. Therefore, a need for a more secure online environment is highly required to obtain the trust and confidence of every user [3]. To ensure security for online services and transactions, cryptography has been severely used since it makes information indecipherable to illegitimate users [4]-[6] thus promoting a safer virtual world [7][8]. Several different types of crypto-algorithms are used to secure data although the traditional algorithms have shown drawbacks in terms of speed, key size, and performance [4], [9].

Cryptographic algorithms or Crypto-algorithms are categorized as either symmetric or asymmetric where [1], [10]-[12] symmetric algorithms, as compared to asymmetric, are extremely fast [8], [13] and secure through strong key size [8]. Among symmetric algorithms, Blowfish is considered superior in terms of security, avalanche effect, throughput, memory usage, execution time, and power consumption [14]. It gives better performance compared to other algorithms [15] and demonstrated excellent security as it is known to have no successful attacks [15]-[17]. However, Blowfish became unfavorable to be used because of little block size support despite being known as an unbroken algorithm. This paved way for its successors, Twofish and Threefish, to be desirably practical to use in many applications [17]. Nevertheless, both Twofish and Threefish offer good security, but, were found to be vulnerable to some attacks [15]-[17].

This study aimed to capitalize on the strengths of Blowfish algorithm while escalating its support to 128-bits block size to improve performance with wider block size text input, improve complexity by using dynamic selection encryption method and reduce its execution of cipher function in randomly determined rounds to further improve the algorithm's execution time and complexity. The results of this study will be a good contribution to the continuous developments in the field of information security, particularly in cryptography.

[+] Corresponding author.
E-mail address: copperstone1999@gmail.com.

2. Literature Review

Blowfish algorithm is a robust general-purpose keyed symmetric block cipher algorithm that supports 64-bit block size and a variable key length from 32 bits up to 448 bits. It is a 16-round Feistel cipher that uses large key-dependent S-boxes and requires 521 iterations to generate all essential subkeys [7], [8], [12]. Blowfish algorithm starts by splitting the 64-bits plain text into two equal blocks followed by bitwise XOR operation between the first 32-bits block (L) and the first 32-bits P array and the computation of the cipher function or function f. The computation of cipher function is considered as the most complex part of this algorithm where S-boxes is utilized [18]. The two 32-bits output segments (L and R) are exchanged and the process repeats for 16 iterations to produce the ciphertext. Because it was not patented and has a free license [7], various modifications of Blowfish to further improve its performance and be suitable for different intended applications has been introduced. Twofish and Threefish algorithms were among the most common algorithms directly known with Blowfish.

Twofish algorithm, derived from Blowfish, SAFER, and DES algorithms, was introduced and became a finalist of NIST to become an AES standard [19]. It is a symmetric crypto-algorithm that supports 128-bits block size input and uses 128, 192, and 256 bits variable key lengths, precomputed key-dependent S-boxes, and a complex key schedule [20]. Although it is considered to possess good security [15], [16], it was found to be vulnerable from some attacks including differential attack and related key attack [15], [17].

The Threefish algorithm is a symmetric key block ciphering algorithm with tweakable block ciphering algorithm directly known with Blowfish and Twofish algorithms. It takes three information sources including the key, a change, and a block of a back rub encoded using change esteem which is 128-bits for all block sizes. Threefish uses 256, 512, and 1024 bits key whose key size is equal to the size of the text block input. It does not use S-box or another table [17], [21]. Threefish offers good security like Twofish but was found to be vulnerable to related key and boomerang attack [17].

Different crypto-algorithms had been investigated by various literature showing its strengths and weakness under different parameters. When Blowfish algorithm and its successors were compared on the basis of diverse parameters, Blowfish claimed to offer high-security level and had shown no successful attacks over the years as compared to its successors.

3. Blowfish-128

Blowfish-128 is a modified version of the Blowfish algorithm designed to support 128-bits block size while capitalizing on its strengths. In this concept, the text input is divided into blocks of equal lengths having 64-bits wide each block. Dynamic selection method determines the encryption process of every block, either with the use of Crypto-algorithm processor or Inverted Crypto-algorithm processor. One (1) block is encrypted using the first crypto-processor and the other by the second crypto-processor distinguished by the sum of generated numbers. The random determination as to which round the cipher function will and will not be executed is also implemented in both processors that are dependent according to generated numbers as well. The combined output of both crypto-processors serves as the encrypted text or the ciphertext. The concept in detail is composed of five components as shown in Figure 1.

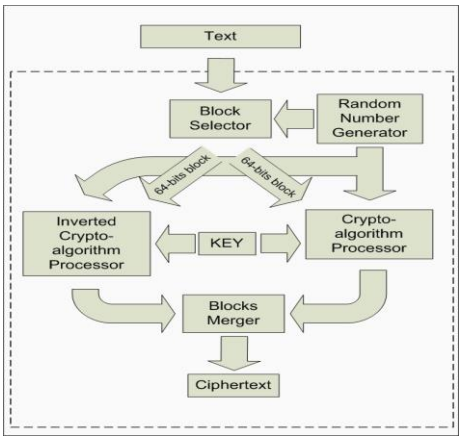


Fig. 1: Proposed concept of the study.

The components of the proposed concept (Figure 1) includes Block Selector, Random Number Generator, Crypto-algorithm Processor, Inverted Crypto-algorithm Processor, and Blocks Merger. The Block Selector divides the input text into equal parts of 64-bits block size. A random selection as to which block goes to the Crypto-algorithm or the Inverted Crypto-algorithm Processors is determined by the Random Number Generator which generates 8 numbers from 1 to 16 and its sum will define which among the processors the blocks has to go through, thus making the encryption selective to improve complexity and increase security. If the sum is odd, the 64-bits left side block undergoes encryption with the Crypto-algorithm Processor while the other 64-bits block is encrypted using the Inverted Crypto-algorithm Processor. Both processors execute the cipher function only on selected rounds demarcated by the Random Number Generator. For the Crypto-algorithm Processor, cipher function executes on rounds included in the randomly generated numbers, whereas, in the Inverted Crypto-algorithm, cipher function executes on rounds not present in the array of randomly generated numbers. This method improves the execution time and the security of the algorithm through reduction of the execution of cipher function into half and through complexity and confusion respectively. To produce the ciphertext, the two outputs of both processors are combined by Blocks Merger. The output of the Blocks Merger is the encrypted text or the ciphertext [22].

4. Results and Discussions

Both Blowfish and Blowfish-128 algorithms were implemented and tested in a Pentium, Dual-Core powered CPU with 4GB of memory and running on Windows 10 64-bit operating system. XAMPP v3.2.1 was utilized to provide a web server solution stack for experimentation purposes. The Blowfish algorithm written by Matt Harris in PHP [23] anchored directly from Schneier's C code example on his website was used as the starting point [24]. An additional class, however, was created to process actual inputs and provides evaluation results for analysis. To verify the validity of the implementation, it was tested using the test vectors provided by Eric Young also posted on Schneier's website [24].

For consistency in running the tests and achieve a fair basis for comparison, the same sets of input data were used all throughout the experimentation and under ECB mode. Categories of 128, 256, 512 and 1024 bits text having ten (10) sets of data in every category and a 64-bit key were used as inputs for encryption and decryption. For the Blowfish-128 implementation, two sets of random numbers were tested with first set including 0, 1, 3, 4, 6, 7, 12, 15 as random numbers and the other set having 1, 2, 3, 5, 6, 9, 12, 15 whose sum are even and odd numbers respectively. The avalanche effect, throughput, and execution time were taken and recorded for every input set for performance evaluation and analysis.

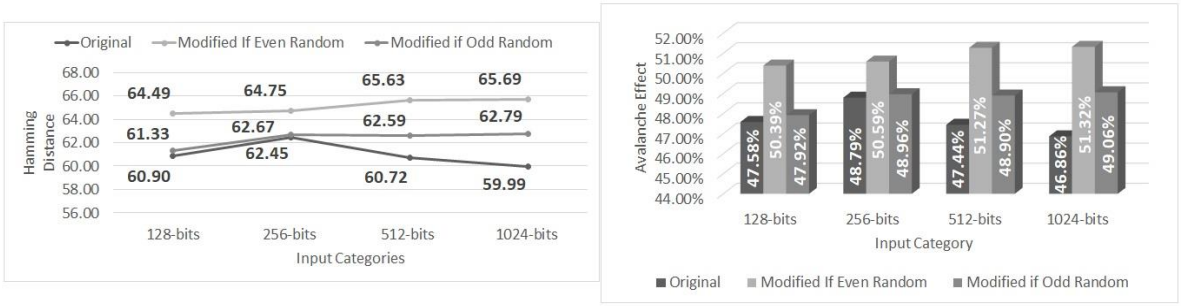
4.1 Avalanche Effect Experimental Results

Avalanche effect is an important property of a cryptographic algorithm since it depicts the strength of an algorithm in terms of diffusion. It is obtained by evaluating the substantial change in the output (ciphertext) when a slight change in input (plain text/key) is applied [7][12][18]. To determine the avalanche effect, equation 1 was used [7].

$$\text{Avalanche Effect} = (\text{Hamming Distance} \div \text{Total Bits Length}) \quad (1)$$

The Hamming distance can be determined by taking the dissimilarity between the ciphertext produced before and after the slight change in the key was applied [7]. In this study, the key's 9th bit was flipped and the difference between the two ciphertexts was recorded using PHP's array_diff_assoc instruction [25]. Presented in Figure 2 are the avalanche effect experimental results obtained when Blowfish and Blowfish-128 algorithms were implemented at even and odd sum of the random numbers and at different input categories using ECB mode. Figure 2(a) shows the recorded hamming distance while Figure 2(b) is the computed avalanche effect using Equation 1. The recorded hamming distance (Figure 2(a)) shows that Blowfish-128, in general, caused an increase in the average number of bits changed as the 9th bit of the key was flipped with the biggest improvement observed for even number sum of the randomly generated numbers. This lead to an improved avalanche effect for Blowfish-128 of utmost 9.52 % and at least 0.35 % improvements as shown in Figure 2(b). The highest recorded improvements in terms of avalanche effect were at 1024-bits input regardless whether the sum of the random numbers was odd or even. A 4.46%

difference between Blowfish and Blowfish-128 implementations with even number sum of random numbers was observed and 2.20% for the odd number. This signifies that Blowfish-128 offers stronger diffusion property compared to Blowfish algorithm.



(a) (b)

Fig. 2: (a) Recorded hamming distance, and (b) computed avalanche effect.

4.2 Throughput Experimental Result

Another important property that illustrates the performance of a crypto-algorithm is throughput. It is considered to be directly proportional to the performance of a crypto-algorithm, thus, higher throughput means higher performance. Throughput is calculated using equation 2 [26].

$$\text{Throughput} = (\text{Size of Text Input} \div \text{Encryption Time}) \tag{2}$$

In this study, Blowfish-128 bids higher throughput as compared to Blowfish regardless whether the sum of random numbers is odd or even (Figure 3). Throughput difference of 369.51 bits per milliseconds can be observed for the even random number while 368 bits per milliseconds difference for the odd random number. This signifies that Blowfish-128 offers better performance in terms of throughput than Blowfish with 37.99 % improvement if the sum of the random number is an even number and 37.83 % improvement for an odd sum of the random number.

Fig. 3: Experimental result in throughput.

4.3 Execution Time Experimental Results

The execution time (equation 3) refers to the total time expended in converting the plain text to the ciphertext (encryption time) and the time necessary to recover the plaintext from the ciphertext (decryption time) [27].

$$\text{Execution Time} = \text{Average Encryption Time} + \text{Average Decryption Time} \tag{3}$$

This impacts the performance of the system and determines how fast the system is and how responsive it is [7]. The average encryption and decryption time of the same sets of inputs executed repeatedly was considered to ensure fair results. It can be noted from Table 1 that Blowfish-128 consistently require less time to encrypt plain text and recover it from a ciphertext as compared to Blowfish regardless whether the

sum of the random number used was odd or even. Focusing on percentage improvement over Blowfish algorithm (Figure 4), it can be seen that the biggest noticeable improvement with Blowfish-128 occurred at 128-bits input where execution time decreased by almost half. Given a 128-bit input, the said modification was 51.55% and 42.40% faster in terms of encryption time, and 26.32% and 17.75% faster decryption time for even and the odd sum of random numbers respectively as compared to Blowfish algorithm. This means that at 128-bit input, Blowfish-128 was 41.68% and 32.76% faster in terms of execution time for an even and odd sum of random numbers respectively. In general, significant increases in execution performance regardless of the input category as a whole was observed.

Table 1: Average encryption and decryption time in milliseconds

		128-bits input		256-bits input		512-bits input		1024-bits input	
		Encryption	Decryption	Encryption	Decryption	Encryption	Decryption	Encryption	Decryption
Blowfish		0.24	0.15	0.26	0.26	0.52	0.45	0.95	0.87
Blowfish-128	Even Random Number	0.12	0.11	0.20	0.20	0.38	0.37	0.74	0.69
	Odd Random Number	0.14	0.13	0.20	0.19	0.37	0.36	0.73	0.69

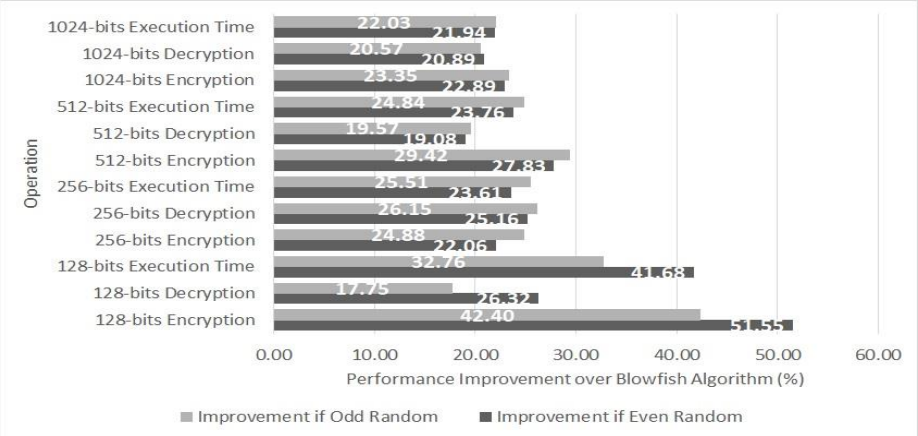


Fig. 4: Experimental result in execution time.

5. Conclusions

In this study, the Blowfish algorithm was implemented by extending its supported block size from 64-bits to 128-bits with the use of dynamic selection encryption method and randomly determined rounds for cipher function execution. The results and analysis conducted in this study revealed that Blowfish-128 had not compromised the features of Blowfish algorithm, instead, had improved its performance in terms of throughput and increased the security concerning avalanche effect at reduced execution time. It had also added an extra layer of security by requiring other data aside from the key to decipher an information such as the encryption method and specific rounds where the cipher function should and should not be executed. These results lead to a conclusion that with Blowfish-128, it will be more difficult and complex for an unauthorized individual to decipher the information even if the secret key is compromised, thus, making the virtual world safer and more resilient to attacks by unauthorized parties.

6. References

[1] Swathi S V, Lahari P M, and Bindu A Thomas, "Encryption Algorithms: A Survey," *International Journal of Advanced Research in Computer Science & Technology*, vol. 4, no. 2, 2016.

[2] A. Joshi, M. Wazid, and R. H. Goudar, "An Efficient Cryptographic Scheme for Text Message Protection Against Brute Force and Cryptanalytic Attacks," *Procedia Computer Science*, vol. 48, pp. 360–366, Jan. 2015.

[3] D. Dasgupta, A. Roy, and A. Nag, "Toward the design of adaptive selection strategies for multi-factor authentication," *Elsevier Journal of Computers & Security*, vol. 63, pp. 85–116, Nov. 2016.

[4] S. K. Pujari, G. Bhattacharjee, and S. Bhoi, "A Hybridized Model for Image Encryption through Genetic

Algorithm and DNA Sequence," *Procedia Computer Science*, vol. 125, pp. 165–171, Jan. 2018.

[5] S. Deng, D. Yue, A. Zhou, X. Fu, L. Yang, and Y. Xue, "Distributed content filtering algorithm based on data label and policy expression in active distribution networks," *Elsevier Journal of Neurocomputing*, vol. 270, pp. 159–169, Dec. 2017.

[6] K. Dhiman and S. S. Kasana, "Extended visual cryptography techniques for true color images," *Elsevier Journal of Computers & Electrical Engineering*, Oct. 2017.

[7] P. Patil, P. Narayankar, Narayan D.G., and Meena S.M., "A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish," *Procedia Computer Science*, vol. 78, pp. 617–624, Jan. 2016.

[8] A. Ramesh and A. Suruliandi, "Performance analysis of encryption algorithms for Information Security," in *2013 International Conference on Circuits, Power and Computing Technologies (ICCPCT)*, 2013, pp. 840–844.

[9] Miss. A.Usha and Dr. A. Subramani, "Performance Study of Key Developer Data Encryption and Decryption Algorithm (KDDEDA) with AES, DES, and BLOWFISH," *International Journal Of Engineering And Computer Science (IJECS)*, vol. 5, no. 12, pp. 19596–19611, 2016.

[10] Ü. Çavuşoğlu, A. Akgül, A. Zengin, and I. Pehlivan, "The design and implementation of hybrid RSA algorithm using a novel chaos-based RNG," *Elsevier Journal of Chaos, Solitons & Fractals*, vol. 104, pp. 655–667, Nov. 2017.

[11] M. Kumar, S. Kumar, R. Budhiraja, M. K. Das, and S. Singh, "A cryptographic model based on the logistic map and a 3-D matrix," *Journal of Information Security and Applications*, vol. 32, pp. 47–58, Feb. 2017.

[12] Youssouf Mahamat koukou, Siti Hajar Othman, Maheyzah MD Siraj, and Herve Nkiama, "Comparative Study Of AES, Blowfish, CAST-128 And DES Encryption Algorithm," *IOSR Journal of Engineering (IOSRJEN)*, vol. 6, no. 6, pp. 1–7, 2016.

[13] A. Bhardwaj, G. V. B. Subrahmanyam, V. Avasthi, and H. Sastry, "Security Algorithms for Cloud Computing," *Procedia Computer Science*, vol. 85, pp. 535–542, Jan. 2016.

[14] M. Suresh and M. Neema, "Hardware Implementation of Blowfish Algorithm for the Secure Data Transmission in the Internet of Things," *Procedia Technology*, vol. 25, no. Supplement C, pp. 248–255, Jan. 2016.

[15] P. Dixit, A. K. Gupta, M. C. Trivedi, and V. K. Yadav, "Traditional and Hybrid Encryption Techniques: A Survey," in *Networking Communication and Data Knowledge Engineering*, Springer, Singapore, 2018, pp. 239–248.

[16] R. Rajan, D. G. Murugaboopathi, and D. C.Parthasarathy, "Analysis and assessment of various cryptographic techniques based on a variety of features," *International Journal of Engineering & Technology*, vol. 7, no. 1.9, pp. 28–33, Mar. 2018.

[17] Shin Woo Jang, "Comparative Analysis of AES, Blowfish, Twofish and Threefish Encryption Algorithms," *AAM*, vol. 10, no. 1–24, 2017.

[18] A. Alabaichi, F. Ahmad, and R. Mahmod, "Security analysis of blowfish algorithm," in *2013 Second International Conference on Informatics Applications (ICIA)*, 2013, pp. 12–18.

[19] M. A. Bahnasawi *et al.*, "ASIC-oriented comparative review of hardware security algorithms for the internet of things applications," in *2016 28th International Conference on Microelectronics (ICM)*, 2016, pp. 285–288.

[20] Y. Harmouch and R. E. Kouch, "A Fair Comparison Between Several Ciphers in Characteristics, Safety and Speed Test," in *Europe and MENA Cooperation Advances in Information and Communication Technologies*, Springer, Cham, 2017, pp. 535–547.

[21] N. At, J. L. Beuchat, and I. San, "Compact Implementation of Threefish and Skein on FPGA," in *2012 5th International Conference on New Technologies, Mobility and Security (NTMS)*, 2012, pp. 1–5.

[22] A. R. L. Reyes, E. D. Festijo, and R. P. Medina, "Securing One Time Password (OTP) for Multi-Factor Out-of-Band Authentication through a 128-bit Blowfish Algorithm," *International Journal of Communication Networks and Information Security (IJCNIS)*, vol. 10, no. 1, Apr. 2018.

[23] Harris, M. (2018). themattharris (Matt Harris). [online] GitHub. Available at: https://github.com/themattharris [Accessed 2 Feb. 2018].

[24] Schneier.com. (2018). Schneier on Security: The Blowfish Encryption Algorithm. [online] Available at: https://www.schneier.com/academic/blowfish/ [Accessed 2 Feb. 2018].

[25] Php.net. (2018). PHP: array_diff_assoc - Manual. [online] Available at: http://php.net/manual/en/function.array-diff-assoc.php [Accessed 2 Feb. 2018].

[26] M. Arora, S. Sharma, and D. Engles, "Parametric comparison of EMDS algorithm with some symmetric cryptosystems," *Egyptian Informatics Journal*, vol. 18, no. 2, pp. 141–149, Jul. 2017.

[27] K. L. Prasad, P. Anusha, G. Jyothi, and K. Dileepkumar, "Design and Analysis of Secure and Efficient Image with Embedded Sensitive Information Transferring Technique using Blowfish Algorithm," *i-Manager's Journal on Information Technology; Nagercoil*, vol. 5, no. 3, pp. 1–8, Aug. 2016.