ISSN 2278-3091
Volume 9, No.3, May - June 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse122932020.pdf
https://doi.org/10.30534/ijatcse/2020/122932020

An Enhanced Nihilist Cipher Using Blum Blum Shub Algorithm

Allemar Jhone P. Delima[1], Jan Carlo T. Arroyo[2]
[1,2]College of Computing Education, University of Mindanao, Davao City, Davao del Sur, Philippines
[1]College of Engineering, Technology and Management, Cebu Technological University-Barili Campus, Cebu, Philippines
allemardelima@umindanao.edu.ph [1], jancarlo_arroyo@umindanao.edu.ph[2]

ABSTRACT

The Nihilist cipher is a symmetric encryption cipher that works by substituting the characters of the plaintext and the keyword and form bigrams based on their character placement and coordinates within the Polybius square. The resulting bigrams from both plaintext and key are added to form the ciphertext. However, the pairing process is unsatisfactory due to its key distribution scheme making the entire process vulnerable to attacks. With this, the Blum Blum Shub (BBS) algorithm is added to randomize the key pairing process of the Nihilist cipher. Therefore, this paper discusses the processes of the proposed cipher methodology based on the combined Nihilist cipher and Blum Blum Shub algorithm. The hybrid methodology offers secure encryption and decryption process, as evident in the simulation results conducted.

Key words: Blum Blum Shub, cryptography, encryption, hybrid ciphers, Nihilist cipher

1. INTRODUCTION

Cryptography [1] has been part of human's daily lives. Most of the digital communications require authentication over open channels such as the Internet as hackers, crackers, eavesdroppers, and other adversaries are keeping an eye on different data available almost everywhere to be used for everything [2]. Governmental and non-governmental agencies, commercial and non-profit enterprises, as well as the general public all rely on security [3], of which the cipher process of cryptography is an intrinsic part. A cipher is a process that transforms data, in the form of plain text into an incomprehensible format called the encryption with a reversing decryption process that transforms ciphertext back into its original plain text format [4].

Some of the commonly used ciphers in the literature are the Nihilist cipher [5], Base64 cipher [6]–[8], Beaufort cipher [4], Bifid cipher [9], Caesar cipher [10]–[12], Enigma Machine cipher [4], Four-square cipher [4], Grille cipher [13], Hill cipher [14], Homophonic substitution cipher [15], [16], and Permutation cipher [4], among others. In this study, the famous Nihilist cipher is modified and added with the Blum Blum Shub algorithm. The Blum-Blum-Shub algorithm is a secure pseudorandom number generator that produces a sequence by reducing squares modulo the product of two Blum primes [17]. The proposed hybridization of the two algorithm aims to produce a more secure ciphertext by enhancing the key generation process of the traditional Nihilist cipher.

2. METHODOLOGY

2.1 Nihilist Cipher

The Nihilist cipher is a substitution cipher that makes use of a matrix to produce ciphertext. In using the Nihilist cipher, the plaintext and the keyword are translated into its numerical form through character substitution using a Polybius square to generate bigrams that represent the coordinates of the character in the grid. The bigrams from the plaintext and keyword are added together to generate the ciphertext [18].

The traditional Nihilist cipher uses a 5x5 grid matrix, as presented in Table 1. The grid is filled with alphabets written from left to right, then from top to bottom.

Table 1: Nihilist cipher table

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I/J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

First, the plaintext is converted to a numeric value by matching each character to the given matrix and retrieving its row-column index known as a bigram. For instance, the plaintext PASSAGE is translated into 35 11 43 43 11 22 15, as presented in Table 2.

Table 2: Plaintext conversion

Plaintext	P	A	S	S	A	G	E
Position	1	2	3	4	5	6	7
Converted Plaintext	35	11	43	43	11	22	15

Second, the keyword is also converted to its numeric equivalent by matching each character to the given matrix and retrieving its equivalent bigrams. For example, the keyword KEY is converted to 25 15 45, as shown in Table 3.

Table 3: Keyword conversion

Plaintext	K	E	Y
Position	1	2	3
Converted Keyword	25	15	45

Further, each character of the keyword is paired with each character of the plaintext. Since the given keyword is composed of only three letters, the characters are repeatedly matched up to the length of the plaintext, as seen in Table 4.

Table 4: Plaintext-keyword pairing

Plaintext	P	A	S	S	A	G	E
Converted Plaintext	35	11	43	43	11	22	15
Keyword	K	E	Y	K	E	Y	K
Converted Keyword	25	15	45	25	15	45	25

Lastly, each pair of plaintext and keyword bigrams are summed to generate the ciphertext. Based on the example, the first character P is encrypted as 60 (35+25), A is encrypted as 26 (11+15), and so forth. Hence, the plaintext PASSAGE is encrypted as 60 26 88 68 26 67 40 using the keyword KEY, as presented in Table 5.

Table 5: Nihilist cipher encryption

Plaintext	P	A	S	S	A	G	E
Converted Plaintext	35	11	43	43	11	22	15
Keyword	K	E	Y	K	E	Y	K
Converted Keyword	25	15	45	25	15	45	25
Final Ciphertext	60	26	88	68	26	67	40

To decrypt the ciphertext, the keyword is first identified. The keyword is converted to its numerical equivalent. This is then deducted from the ciphertext value to generate the bigrams for the plaintext. Each resulting bigram is matched with the Polybius square to retrieve the plaintext.

One advantage of using the Nihilist cipher over the Polybius cipher is that the former may generate varied ciphertext values for identical characters as opposed to the latter, which produces the same values for identical characters. For instance, the character S is encrypted as 88 or 68 and E as 30 or 40, as seen on the results above.

Cryptanalysis for Nihilist cipher is done through pattern analysis and factoring. The keyword length can be guessed by looking at low and high number patterns in the ciphertext. This is more obvious if a lengthy plaintext is used. The keyword may also be guessed by factoring, such that the ciphertext 89 can only be made by the factors 45+44 based on the limited combination of 1, 2, 3, 4, 5 from the matrix. This is due to the simple addition approach of the repeating keywords paired with the plaintext. For example, in a 5x5 matrix, if a ciphertext value is more than 100, then that would easily mean that both plaintext and keyword values come from the 5th row of the Polybius square.

2.2 Blum Blum Shub Algorithm

Blum Blum Shub (BBS) is a well-known probabilistically secure pseudo-random number generator (PRNG) proposed by Lenore Blum, Manuel Blum, and Michael Shub in 1986 [19]. The BBS algorithm produces random numbers by using two primes p and q, such that p and $q \equiv 3 \pmod 4$ and $gcd(p,q) = 1$. These primes, together with a random seed, is computed as:

$$b_{n+1} = b_n^2 \bmod M \quad (1)$$

where:

(i) b is a random seed value
(ii) M is the product of two large prime numbers p and q
(iii) p and q are congruent to $3 \pmod 4$

The example below shows how BBS generates random numbers:

- Let, $p = 7 \equiv 3 \pmod 4$
  Let $q = 19 \equiv 3 \pmod 4$
  Then, n = p * q = 7 * 19 = 133
- Choose a seed value $b_0 = 100$
- $b_1 = 100^2 \pmod{133} = 25$
- $b_2 = 25^2 \pmod{133} = 93$
- $b_3 = 93^2 \pmod{133} = 4$
- *so forth...*

2.3 Proposed Cipher Process

The proposed process introduces the use of BBS and the XOR operation to further enhance the security of the cipher. The addition of BBS and XOR increases the randomness of the cipher to minimize patterns in the ciphertext values. The proposed process also extends the Nihilist cipher into a 6x6 matrix to include digits in the range of allowable characters to be encoded and decoded. The encryption process is presented in Figure 1, while the decryption process is presented in Figure 2.

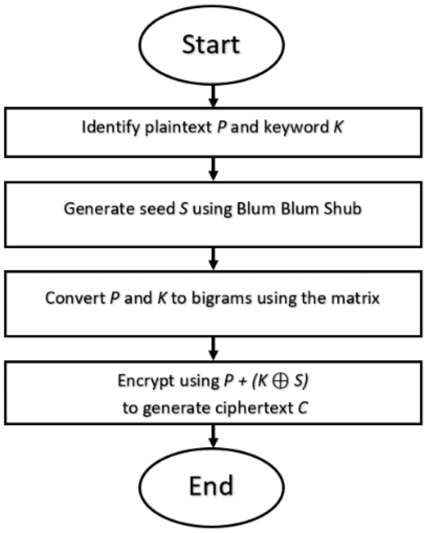

Figure 1: Encryption process

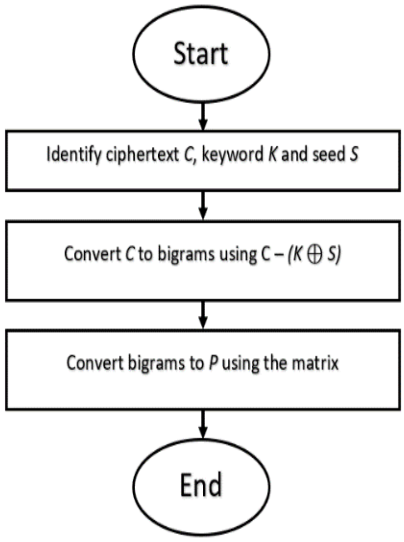

Figure 2: Decryption process

Encryption using the proposed method involves a plaintext, keyword, and randomly generated seed. For instance, the given plaintext is PASSAGE is used where the keyword is KEY, and the seed is 7 0 7 3 1 0 2. The plaintext and the keyword are translated into bigrams using the 6x6 matrix. Further, each character from the keyword is paired with each character from the plaintext. This is done repeatedly until the length of the plaintext. Each seed value is paired as well. A sample 6x6 matrix is presented in Table 6, while the converted values are presented in Table 7.

Table 6: A 6x6 nihilist cipher table

	1	2	3	4	5	6
1	A	B	C	D	E	F
2	G	H	I	J	K	L
3	M	N	O	P	Q	R
4	S	T	U	V	W	X
5	Y	Z	0	1	2	3
6	4	5	6	7	8	9

Table 7: Plaintext and keyword conversion

Plaintext	P	A	S	S	A	G	E
Converted Plaintext	35	11	43	43	11	22	15
Keyword	K	E	Y	K	E	Y	K
Converted Keyword	25	15	45	25	15	45	25
Seed	7	0	7	3	1	0	2

The next step involves performing addition and XOR operations. Each seed value is XORed with the corresponding keyword bigram and then added to the plaintext bigram using the equation $P + (K \oplus S)$. Based on the given example, the first character M is encrypted as 64 = (35 + (25 $\oplus$ 7)), E is encrypted as 26 = (11 + (15 $\oplus$ 0), and so forth. The plaintext PASSAGE is now encrypted as 64 26 93 67 25 72 42 using the keyword KEY, as seen in Table 8.

Table 8: Nihilist cipher encryption

Plaintext	P	A	S	S	A	G	E
Converted Plaintext	35	11	43	43	11	22	15
Keyword	K	E	Y	K	E	Y	K
Converted Keyword	25	15	45	25	15	45	25
Final Ciphertext	64	26	93	67	25	72	42

The decryption process requires access to the ciphertext, keyword, and seed values. The seed is XORed with the keyword bigram and deducted from the plaintext bigram. Each resulting bigram is now matched with the matrix to retrieve the plaintext.

3. RESULTS AND DISCUSSION

In order to assess the viability of the proposed method, two test cases were conducted using a variety of plaintext and keys. In this study, a 6x6 matrix was used to test both the standard and modified Nihilist ciphers.

In test case 1, a 43-byte plaintext is encrypted using the keyword DOG, as shown in Table 9. Each encrypted ciphertext for the standard and enhanced Nihilist cipher is checked for its low and high patterns. Based on the results of the standard Nihilist cipher, a cryptanalyst may easily recognize the keyword pattern LOW HIGH LOW (d-o-g) in the ciphertext because of how frequent it appears (6 times). Cryptanalysts may also use factoring to easily identify the keyword bigram in the standard Nihilist cipher, such that the first ciphertext bigram 27 can only be produced using the digits 11+16, 12+15, or 14+13 when using the matrix. The enhanced process performs better as the identified LOW HIGH LOW pattern is not as apparent since it only appears three times. On the other hand, attackers may not use factoring to determine the keyword bigram because the matrix could not generate bigrams lower than 11 and higher than 66.

Table 9: Test case 1

Plaintext	cryptographyisthescienceofhidinginformation
Size	43 bytes
Keyword	Dog
Seed	5 4 3 3 5 0 7 5 6 1 1 3 0 6 4 7 7 7 2 4 7 2 2 4 2 6 2 6 4 5 3 3 7 3 2 0 6 6 6 3 3 7 2
Nihilist Ciphertext	27 69 72 48 75 54 35 69 32 48 55 72 37 74 63 36 48 62 27 56 36 46 46 36 47 49 43 37 47 44 46 54 44 46 49 54 50 64 32 56 56 54 46
Pattern	Low High High Low High Low Low High Low High High High Low High Low Low High High Low High Low High Low Low High High Low Low High Low High High Low High High High Low High Low High Low Low Low
Modified Nihilist Ciphertext	243 57 200 48 115 186 35 245 96 208 199 214 225 202 63 26 176 60 211 182 90 46 50 98 175 69 155 93 175 74 208 184 194 176 181 184 172 64 32 166 184 54 206
Pattern	Low Low High Low High High Low High Low High Low High High Low Low Low High Low High Low Low Low High High High Low High Low High Low High Low High Low High High Low Low Low High High Low High

In test case 2, a 77-byte plaintext is encrypted using the keyword INFO, as shown in Table 10. Each encrypted ciphertext generated by the standard and enhanced Nihilist cipher is checked for its low and high patterns. Based on the results generated by the standard Nihilist cipher, a cryptanalyst may easily recognize the keyword pattern LOW HIGH LOW HIGH (i-n-f-o) in the ciphertext because of how frequent it appears (8 times). Cryptanalysts may also use factoring to easily identify the keyword bigram in the standard Nihilist cipher, such that the ciphertext bigram 31 can only be produced using the digits 16+15 when using the matrix. The enhanced process performs better as the identified LOW HIGH LOW pattern is not as apparent since it only appears two times. Also, attackers may not use factoring to determine the keyword bigram because the matrix could not generate bigrams lower than 11 and higher than 66.

Table 10: Test Case 2

Plaintext	TheNihilistcipherisamonoalphabeticclassicalcipherusedbytheRussiansagainstCzar
Size	77 bytes
Keyword	Info
Seed	77 105 9 50 120 27 38 9 60 53 113 73 9 51 16 69 93 27 121 2 49 16 47 105 79 15 37 54 104 44 78 90 114 115 94 39 52 111 127 65 6 45 43 12 26 119 72 61 1 2 58 99 94 66 21 92 85 85 31 25 111 46 47 111 83 95 32 120 7 45 57 32 35 108 121 15 89
Nihilist Ciphertext	65 54 31 65 46 54 39 59 46 73 58 46 46 66 38 48 59 55 57 44 54 65 48 66 34 58 50 55 34 44 31 75 46 45 29 59 34 73 57 56 36 43 42 46 46 66 38 48 59 75 57 48 37 44 67 75 45 47 52 76 64 73 39 44 55 73 27 54 34 55 48 74 65 45 68 44 59
Pattern	Low Low Low High Low High Low High Low High Low Low Low High Low High High Low High Low High High Low High Low High Low High Low High Low High Low Low Low High Low High Low Low Low High Low High Low High Low High High High Low Low Low High High High Low High High High Low High Low High High High Low High Low High Low High Low Low High Low High
Modified Nihilist Ciphertext	132 95 40 51 134 81 77 66 66 62 139 117 53 53 22 115 110 82 146 46 69 81 95 105 99 73 87 45 138 24 109 165 124 96 91 32 46 120 152 119 30 24 85 58 36 121 110 43 58 77 83 81 87 110 56 167 88 132 51 99 161 55 86 89 100 168 59 110 27 36 73 42 94 89 157 57 114
Pattern	Low Low Low High High Low Low Low High Low High Low Low High Low High Low Low High Low High High High High Low Low High Low High Low High High Low Low Low Low High High High Low Low Low High Low Low High Low Low High High High Low High High Low High Low High Low High High Low High High High High Low High Low High High Low High Low High Low High

4. CONCLUSION

In this paper, the use of the extended Nihilist cipher is introduced to increase the cipher capability of the Nihilist cipher as recommended in the study of [20]. Further, the use of the Blum Blum Shub algorithm, together with the XOR process prior to the generation of ciphertext, increases the cipher's randomness, concealing the low and high number patterns and as well as preventing attackers on guessing the keyword used through factoring. The proposed method shows ciphertext variability and pattern obscureness, making the encrypted data difficult to break.

REFERENCES

[1] W. Stallings, *Cryptography and Network Security Principles and Practices*. Prentice Hall, 2015.

[2] S. Dey, J. Nath, and A. Nath, "An Integrated Symmetric Key Cryptographic Method – Amalgamation of TTJSA Algorithm, Advanced Caesar Cipher Algorithm, Bit Rotation and Reversal method: SJA Algorithm," *Int. J. Mod. Educ. Comput. Sci.*, vol. 4, no. 5, pp. 1–9, 2012. https://doi.org/10.5815/ijmecs.2012.05.01

[3] A. Rayarapu, A. Saxena, N. V. Krishna, and D. Mundhra, "Securing Files Using AES Algorithm," *Int. J. Comput. Sci. Inf. Technol.*, vol. 4, no. 3, pp. 433–435, 2013.

[4] M. S. Hossain Biswas *et al.*, "A systematic study on classical cryptographic cypher in order to design a smallest cipher," *Int. J. Sci. Res. Publ.*, vol. 9, no. 12, pp. 507–11, 2019. https://doi.org/10.29322/IJSRP.9.12.2019.p9662

[5] E. V. Haryannto, M. Zulfadly, Daifiria, M. B. Akbar, and I. Lazuly, "Implementation of Nihilist Cipher Algorithm in Securing Text Data with Md5 Verification," *J. Phys. Conf. Ser.*, vol. 1361, no. 012020, 2019.

[6] F. Anwar, E. H. Rachmawanto, C. A. Sari, and de Rosal Ignatius Moses Setiadi, "StegoCrypt Scheme using LSB-AES Base64," in *International Conference on Information and Communications Technology, ICOIACT 2019*, 2019, pp. 85–90.

[7] A. R. Pathak, S. Deshpande, and M. Panchal, "A Secure Framework for File Encryption Using Base64 Encoding," in *Computing and Network Sustainability*, vol. 75, Springer Singapore, 2019, pp. 359–366. https://doi.org/10.1007/978-981-13-7150-9_38

[8] R. Rahim, S. Sumarno, M. T. Multazam, S. Thamrin, and S. H. Sumantri, "Combination Base64 and GOST algorithm for security process," *J. Phys. Conf. Ser.*, vol. 1402, 2019.

[9] A. Borodzhieva, "MATLAB-based software tool for implementation of Bifid Ciphers," in *International Conference on Computer Systems and Technologies*, 2017, pp. 326–333. https://doi.org/10.1145/3134302.3134333

[10] A. Singh and S. Sharma, "Enhancing Data Security in Cloud Using Split Algorithm, Caesar Cipher, and Vigenere Cipher, Homomorphism Encryption Scheme," in *Emerging Trends in Expert Applications and Security*, 2019, vol. 841, pp. 157–166.

[11] I. Gunawan, Sumarno, H. S. Tambunan, E. Irawan, H. Qurniawan, and D. Hartama, "Combination of Caesar Cipher Algorithm and Rivest Shamir Adleman Algorithm for Securing Document Files and Text Messages," *J. Phys. Conf. Ser.*, vol. 1255, 2019. https://doi.org/10.1088/1742-6596/1255/1/012077

[12] D. Gautam, C. Agrawal, P. Sharma, M. Mehta, and P. Saini, "An Enhanced Cipher Technique Using Vigenere and Modified Caesar Cipher," in *2nd International Conference on Trends in Electronics and Informatics, ICOEI 2018*, 2018.

[13] J. Liu *et al.*, "The Reincarnation of Grille Cipher: A Generative Approach," *Cryptogr. Secur.*, pp. 1–27, 2018.

[14] P. E. Coggins and T. Glatzer, "An Algorithm for a Matrix-Based Enigma Encoder from a Variation of the Hill Cipher as an Application of 2 × 2 Matrices," *Primus*, vol. 30, no. 1, 2020. https://doi.org/10.1080/10511970.2018.1493010

[15] M. Shumay and G. Srivastava, "PixSel: Images as book cipher keys an efficient implementation using partial homophonic substitution ciphers," *Int. J. Electron. Telecommun.*, vol. 64, no. 2, pp. 151–158,

2018.
[16] G. Zhong, "Cryptanalysis of Homophonic Substitution Cipher Using Hidden Markov Models," 2016.
[17] B. Shrestha, "Multiprime Blum-Blum-Shub pseudorandom number generator," 2016.
[18] D. Kahn, *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times to the Internet*. Scribner, 1996.
[19] L. Blum, M. Shub, and M. Blum, "Simple Unpredictable Pseudo-Random Number Generator," *SIAM J. Comput.*, vol. 15, no. 2, pp. 364–383, 1986. https://doi.org/10.1137/0215025
[20] T. S. Kondo and L. J. Mselle, "An Extended Version of the Polybius Cipher," *Int. J. Comput. Appl.*, vol. 79, no. 13, pp. 30–33, 2013. https://doi.org/10.5120/13803-1836