ISSN 2347 - 3983
Volume 8. No. 7, July 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter19872020.pdf
https://doi.org/10.30534/ijeter/2020/19872020

Caesar Cipher with Goldbach Code Compression for Efficient Cryptography

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

ABSTRACT

This paper proposes an enhancement on the traditional Caesar cipher by introducing the concept of compression using the Goldbach code algorithm to conceal the ciphertext generated by the Caesar cipher for added security. The Caesar cipher is one of the early cryptographic algorithms that is widely used for information hiding and data security. However, the traditional Caesar cipher is vulnerable to attacks and is extremely easy to crack. For the Caesar cipher, each character of a message is always replaced by the same fixed character that has been predetermined. To improve the cipher's security feature, the result of the Caesar cipher is encrypted and is compressed using the Goldbach code algorithm. Simulation results revealed that the proposed method produces a less predictable and shortened ciphertext length than the existing process, ensuring a secure and cost-effective encryption process.

Key words: Caesar cipher, cryptography, data security, goldbach code algorithm, hybrid algorithms

1. INTRODUCTION

Data security and privacy have long been one of the essential aspects of a person's life that needs to be considered today as one cannot communicate securely anymore. There is always a threat for data security from the presence of eavesdroppers, hackers, crackers, and the likes, which causes leak of information. Cryptography deals with obscuring data making information more secure [1]. Cryptography offers not only protection from data alteration and theft but also provides security through authentication [2]. Cryptography is of two types: symmetric and asymmetric. The concept where one key is used for ciphering is called symmetric key cryptography. Symmetric key cryptography offers easy encryption and decryption method as there is only one key concerned in the process. However, the key should not be revealed to anybody as it can be blatantly used for the proliferation of data. As opposed to the first method, the asymmetric key cryptography uses a public and private key for encryption and decryption processes [3]. Figures 1 and 2 shows the graphical representation of the encryption and decryption key schemes for both symmetric and asymmetric key cryptography.

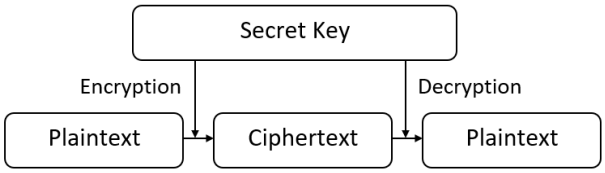

Figure 1: Symmetric key cryptography

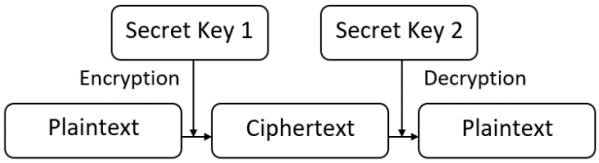

Figure 2: Asymmetric key cryptography

The Caesar cipher is one of the simplest and one of the earliest known symmetric ciphers [4]–[6], aside from Homophonic substitution cipher [7], [8], Hill cipher [9], Grille cipher [10], and more. Caesar cipher is a type of substitution cipher, where the letter in the plaintext is shifted by a certain number of places down the series of the alphabet. However, the Ceasar cipher suffers a problem in ciphertext generation as its output is vulnerable to attacks. In this algorithm, each character of a message is always replaced by the same fixed character that has been predetermined. Hence, a pattern is evident in the ciphertext, making the plaintext unsecured. To establish a more efficient cryptography, the use of Goldbach code algorithm on the ciphertext produced by the traditional Caesar cipher is observed.

2. METHODOLOGY

2.1 Caesar Cipher

One of the most widely known cipher algorithm is the Caesar cipher. This substitution type of cipher replaces every letter in the plaintext with a letter from a fixed number of positions down the alphabet. The encryption is represented using modular arithmetic, where thorough discussion is found at [11]. For the encryption, every character in the plaintext is substituted with another character based on a shifted or rotated cipher alphabet. For example, A is substituted as X, and B is substituted as Y, as shown in Table 1. Decryption is done by reversing the process.

Table 1: Caesar cipher table

Caesar Cipher	ABCDEFGHIJKLMNOPQRSTUVWXYZ
Shifted Caesar Cipher	XYZABCDEFGHIJKLMNOPQRSTUVW

2.2 Goldbach Code

The Goldbach class of codes was developed in 2001 by Peter Fenwick based on Christian Goldbach's conjecture, which states that every even integer greater than four may be expressed as the sum of two odd prime integers [12], [13]. For example, 14 = 3 + 11 and 24 = 11 + 13. Fenwick introduced the simple prime number code called G0. This number system encodes an integer twice with an offset value. Each integer n is encoded with equation *2(n+3)* to generate an equivalent codeword [13]. To identify the equivalent Goldbach G0 codeword for an integer, its two primes are mapped with an array of prime numbers. For instance, let *P*=[3,5,7,11,13,17,19,23,29,31] be the array of the first 10 prime numbers and let *I*=[0,0,0,0,0,0,0,0,0,0] be the array as map for the indices of the primes. Suppose the digit 7 is to be encoded in G0. First, set $n = 7$ and compute for $2(n+3)$, such that $2(7+3) = 20$. Further, identify from *P* the first two distinct primes, which can be summed to get the value 20. Based on *P*, the first two primes for 20 are 7 and 13. These two prime values are mapped to *I* according to their relative indices represented by the value 1, such that *I*=[0,0,1,0,1,0,0,0,0,0]. The codeword is identified by removing trailing zeroes from *I*; thus, the equivalent codeword for the digit 7 is 00101. A sample G0 code for the first 15 integers is shown in Table 2.

Table 2: Goldbach G0 Code

Value *n*	Encode 2(*n*+3)	Sum of Primes	Equivalent Codeword
1	8	3 + 5	11
2	10	3 + 7	101
3	12	5 + 7	011
4	14	3 + 11	1001
5	16	5 + 11	0101
6	18	7 + 11	0011
7	20	7 + 13	00101
8	22	5 + 17	010001
9	24	11 + 13	00011
10	26	7 + 19	0010001
11	28	11 + 17	000101
12	30	13 + 17	000011
13	32	13 + 19	0000101
14	34	11 + 23	00010001
15	36	5 + 31	100000001

2.3 Proposed Cipher Process

The proposed process involves the use of the Caesar cipher for encryption, where the generated ciphertext is compressed using the Golbach G0 code. To encrypt using the proposed process, the steps presented in Figure 3 are executed where detailed steps are as follows:

a. Identify a plaintext and encrypt using the Caesar cipher.
b. Using the ciphertext, count the frequency of each character.
c. Arrange the characters according to the frequency and its order of appearance. The most frequent character shall be first in the list to be encoded in G0 represented by *n*=1.
d. For each character in the ciphertext, compute for *2(n+3)* and find the first two primes of the result.
e. Map the two primes to the list of prime numbers > 2 to retrieve the equivalent codeword in binary format.
f. Repeat steps d and e until all characters are in the G0 code format.
g. List all generated G0 codes according to the original ciphertext.
h. Group the list into 8-bits. Add trailing zeroes if the last group contains less than 8-bits.
i. Convert each group into ASCII to produce the compressed ciphertext.

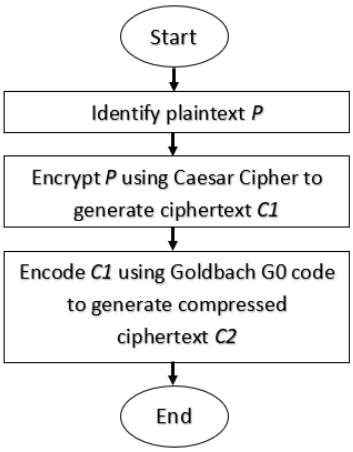

Figure 3: Encryption process

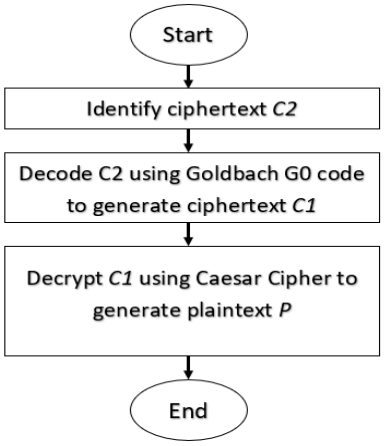

Figure 4: Decryption process

To decrypt using the proposed process, the steps are presented in Figure 4 with detailed steps as follows:

a. Identify a compressed ciphertext value.
b. Convert each ciphertext character to binary format according to its ASCII equivalent.
c. Match each group of bits into the Goldbach G0 code to retrieve ciphertext.
d. Decrypt ciphertext using Caesar cipher to produce plaintext.

3. RESULTS AND DISCUSSION

The proposed method is tested using a variety of plaintext and key in order to test its efficacy. The following test cases are shown in Tables 3-6. In test case 1, the plaintext INFORMATION is encrypted using the proposed cipher process. First, the plaintext is encrypted using the Caesar cipher to generate the initial ciphertext NSKTWRFYNTS. Next, the frequency of each ciphertext character is tallied. The Characters are mapped in the G0 table and are sorted according to its frequency, and its order of appearance in the plaintext, as shown in Table 3.

Table 3: G0 code for test case 1

Character	Frequency	Value *n*	Encode 2(*n*+3)	Sum of Primes	Equivalent Codeword
N	2	1	8	3 + 5	11
S	2	2	10	3 + 7	101
T	1	3	12	5 + 7	011
K	1	4	14	3 + 11	1001
W	1	5	16	5 + 11	0101
R	1	6	18	7 + 11	0011
F	1	7	20	7 + 13	00101
Y	1	8	22	5 + 17	010001

Given the generated ciphertext NSKTWRFYNTS, each character is matched to its equivalent codeword, resulting in 11 101 1001 011 0101 0011 00101 010001 11 011 101. Next, a group of 8-bits is formed using the binary sequence. Further, each group is converted into its equivalent ASCII code character in order to generate the final compressed ciphertext. The simulation results of the process are shown in Table 4.

Table 4: Test case 1 results

Plaintext	INFORMATION
Length	11 characters
Caesar Cipher (5 shifts)	NSKTWRFYNTS
Proposed Method	ìµ2£º
Length	5 characters

In test case 2, the plaintext AABBAABBAABB is encrypted using the proposed cipher process. First, the plaintext is converted into ciphertext FFGGFFGGFFGG using the Caesar cipher. Next, the frequency count of each ciphertext character is identified. The results are mapped and sorted in the G0 table according to the character frequency and the order of appearance in the plaintext, as shown in Table 5.

Table 5: G0 code for test case 2

Character	Frequency	Value *n*	Encode 2(*n*+3)	Sum of Primes	Equivalent Codeword
F	6	1	8	3 + 5	11
G	6	2	10	3 + 7	101

Every character of the ciphertext FFGGFFGGFFGG is substituted with its equivalent codeword, resulting in 11 11 101 101 11 11 101 101 11 11 101 101. Furthermore, the binary sequence is divided into groups of 8-bits and is converted into its equivalent ASCII code character in order to generate the final compressed ciphertext. The results are shown in Table 6.

Table 6: Test case 2 results

Plaintext	AABBAABBAABB
Length	12 characters
Caesar Cipher (5 shifts)	FFGGFFGGFFGG
Proposed Method	û~ß´
Length	4 characters

Based on the findings in the test cases, the proposed method not only masks the plaintext but also shortens its length, thus, making it a cost-efficient cryptography scheme. The proposed method also overcomes the weakness of substitution ciphers, such as the Caesar cipher, wherein the resulting ciphertext does not show apparent patterns for its substituted values for identical characters in the plaintext as presented in test case 2.

4. CONCLUSION

The security of data has been achieved with the use of a hybrid Caesar cipher and Goldbach code algorithm. Simulation results show that the ciphertext generated using the traditional Caesar cipher is made more secure with the concept of compression using the Goldbach algorithm, thus, addressing the vulnerability problem of the traditional Caesar cipher in terms of producing ciphertext with obvious patterns.

REFERENCES

[1] S. N. Gowda, "Innovative enhancement of the Caesar cipher algorithm for cryptography," in *International Conference on Advances in Computing, Communication and Automation*, 2016. https://doi.org/10.1109/ICACCAF.2016.7749010

[2] M. Abdalla, J. H. An, M. Bellare, and C. Namprempre, "From identification to signatures via the Fiat-Shamir transform: Necessary and sufficient conditions for security and forward-security," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3631–3646, 2008. https://doi.org/10.1109/TIT.2008.926303

[3] O. E. Omolara, A. I. Oludare, and S. E. Abdulahi, "Developing a Modified Hybrid Caesar Cipher and Vigenere Cipher for Secure Data Communication," *Comput. Eng. Intell. Syst.*, vol. 5, no. 5, pp. 34–64, 2014.

[4] 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.

[5] 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.

[6] 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. https://doi.org/10.1109/ICOEI.2018.8553910

[7] 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.

[8] G. Zhong, "Cryptanalysis of Homophonic Substitution Cipher Using Hidden Markov Models," 2016.

[9] 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.

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

[11] S. G. Srikantaswamy and H. D. Phaneendra, "Improved Caesar Cipher with Random Number Generation Technique and Multistage Encryption," *Int. J. Cryptogr. Inf. Secur.*, vol. 2, no. 4, pp. 39–49, 2012. https://doi.org/10.5121/ijcis.2012.2405

[12] M. A. Budiman and D. Rachmawati, "On Using Goldbach G0 Codes and Even-Rodeh Codes for Text Compression on Using Goldbach G0 Codes and Even- Rodeh Codes for Text Compression," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 180, 2017.

[13] P. Fenwick, "Variable-Length Integer Codes Based on the Goldbach Conjecture, and Other Additive Codes," *IEEE Trans. Inf. Theory*, vol. 48, no. 6, pp. 2412–2417, 2002. https://doi.org/10.1109/TIT.2002.800483