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/ijeter06872020.pdf
https://doi.org/10.30534/ijeter/2020/06872020

A Keystream-Based Affine Cipher for Dynamic Encryption

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

In this paper, the use of a random seed is integrated to Affine cipher. The random seed is used to generate unique keystream values that dynamically changes the cipher's additive key for every character encrypted. The modification enables the cipher algorithm to produce ciphertext with no trace of repetitive character despite the advent of repetition of characters in the plaintext. Simulation results revealed that the proposed method produces more randomize ciphertext characters as against the traditional Affine cipher. The new method enhances the cipher's capability and complexity in masking plaintext which has paved the way to a more secure and dynamic data encryption.

Key words: Affine cipher, character repetition, cryptography, dynamic encryption, keystream

1. INTRODUCTION

There are numerous ways of securing sensitive information. One method is through encryption or the transformation of data into unintelligible format. Data is secured based on the cryptographic and cipher algorithm [1] used according to the type of data being hidden. Cipher [2] technology can be based on mathematical theories and some are based on classical calculations [3]. In this paper, the classical cipher called Affine cipher [2], [4]–[6] is modified to minimize the production of repetitive characters in the ciphertext. This is realized by introducing a random seed that produces unique encryption keys called keystream for the affine encryption and decryption function.

2. METHODOLOGY

2.1 Affine Cipher

The word affine is a term used to refer to the linear function $f(x) = (ax + b)$, where b is a nonzero value. In cryptography, the Affine cipher is a monoalphabetic substitution cipher based on the Caesar cipher and is defined by the formula $A_{j,d}$: $x \rightarrow y = A_{j,d}(x) = (jx + d) \bmod m$, where m is the range of alphabets, and j and d are the keys [7]. The values for j and m must be coprime so that decryption is possible through the equation $A_{j,d}(y) \equiv j^{-1}(y - d) \bmod m$, where j^{-1} is the inverse modular multiplicative of modulo m that satisfies that equation $1 = aa^{-1} \bmod m$ [8]–[10].

Affine cipher works by mapping a set of alphabets to a range of integers. Using modular arithmetic, each plaintext character is transformed into an integer and that which is transformed into a ciphertext character [8], [9].

For instance, the plaintext UNNEEDED is encrypted using the traditional Affine cipher. First, each character is converted to its numerical equivalent according to its alphabetical index, such that A is 0 and Z is 25. The alphabets A to Z and their corresponding index values are presented in Table 1. Based on the given, the numerical equivalent of the plaintext UNNEEDED represented as x is 20 13 13 4 4 3 4 3, as shown in Table 2.

Table 1: Alphabet indices

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

Table 2: Plaintext numerical equivalent

Plaintext	U	N	N	E	E	D	E	D
x	20	13	13	4	4	3	4	3

Given the affine encryption function $A_{j,d}(x) = (5x + 8) \bmod 26$ where the value 5 is coprime of the modulo, x is the numeric equivalent of the plaintext character, 8 is an arbitrary value for the number of shifts, and modulo 26 is the size of the alphabet, the plaintext is translated to 4 21 21 2 2 23 2 23 as presented in Table 3. These values are converted to ciphertext using the affine table as shown in Table 4.

Table 3: Encryption using Affine cipher

Plaintext	U	N	N	E	E	D	E	D
x	20	13	13	4	4	3	4	3
(5x + 8) mod 26	4	21	21	2	2	23	2	23
Ciphertext	E	V	V	C	C	X	C	X

The decryption process uses the equation $D(y) = 21(y - 8) \bmod 26$ where 21 is the modular multiplicative inverse a^{-1} of modulo 26, y is numeric equivalent of the ciphertext character, and 8 is the number of shifts. For instance, the ciphertext EVVCCXCX is translated as 4 21 21 2 2 23 2 23 and decrypted as UNNEEDED as presented in Table 5.

Table 4: Affine table based on $A_{j,d}(x) = (5x + 8) \bmod 26$

Alphabet	Index	(5x+8) mod 26	Ciphertext
A	0	8	I
B	1	13	N
C	2	18	S
D	3	23	X
E	4	2	C
F	5	7	H
G	6	12	M
H	7	17	R
I	8	22	W
J	9	1	B
K	10	6	G
L	11	11	L
M	12	16	Q
N	13	21	V
O	14	0	A
P	15	5	F
Q	16	10	K
R	17	15	P
S	18	20	U
T	19	25	Z
U	20	4	E
V	21	9	J
W	22	14	O
X	23	19	T
Y	24	24	Y
Z	25	3	D

Table 5: Decryption using Affine cipher

Ciphertext	E	V	V	C	C	X	C	X
y	4	21	21	2	2	23	2	23
21(y-8) mod 26	20	13	13	4	4	3	4	3
Plaintext	U	N	N	E	E	D	E	D

Like any other substitution ciphers, the Affine cipher produces obvious ciphertext patterns for plaintexts containing identical characters. As seen in Table 5, plaintext characters N and E appeared multiple times, thus, as soon as a certain repeating character is decrypted, the remaining identical characters can easily be substituted even without computation or cryptanalysis.

2.2 Proposed Cipher Process

The proposed process extends the capability of the standard Affine cipher by adding the digits 0 to 9 to the range of characters which can be processed, allowing a total of 36 possible plaintext and ciphertext values. The proposed method solves the weakness of the substitution cipher by ensuring that unique ciphertext values are produced and no obvious patterns appear for plaintext composed of identical characters such as AAAAAA or ABABABAB when encrypted. This is achieved by dynamically changing the cipher's additive key for every character encrypted. The modified method uses a random seed value to produce a stream of unique encryption keys through the quadratic function $y = ax^2 + bx + c$, where b is the random seed value; c is the character position; and a is the sum of x, b, and c. The keystream is used as the value d in the encryption and decryption functions.

The modified Affine cipher uses the equation $E(x) = (jx + d) \bmod 36$, where j must be a coprime of $\bmod 36$, x is the character index, and d is a value from the unique keystream. Decryption is done using the equation $D(y) = j^{-1} (y - d) \bmod 36$, where j^{-1} is the modular multiplicative inverse of $\bmod 36$, x is the character index, and d is a value from the unique keystream. The encryption and decryption processes of the modified Affine cipher is shown in Figures 1 and 2.

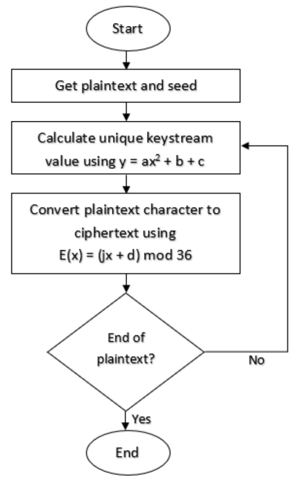

Figure 1: Modified Affine cipher encryption process

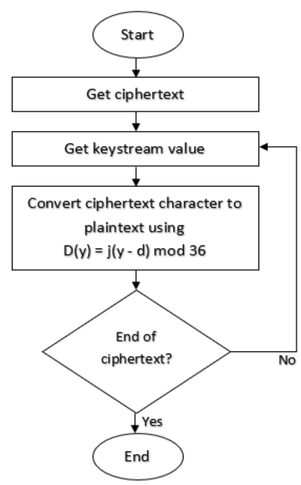

Figure 2: Modified Affine cipher decryption process

To perform encryption using the modified Affine cipher, the following detailed processes are executed:

a. Identify the plaintext value and the random seed. The plaintext is any string composed of letters A to Z and digits 0 to 9, while the random seed can be any integer value. For example, the plaintext is MESSAGE and the seed is 12.
b. Identify plaintext character index in the alphabet represented by x as shown in Table 6.

Table 6: Plaintext character indices

Plaintext	M	E	S	S	A	G	E
Position *c*	1	2	3	4	5	6	7
Index *x*	12	4	18	18	0	6	4

c. Generate keystream values using the equation $d = ax^2 + bx + c$, where *b* is the random seed value; *c* is the character position; and *a* is the sum of *x*, *b*, and *c*. Results are shown in Table 7.

Table 7: Keystream value

Plaintext *p*	M	E	S	S	A	G	E
Position *c*	1	2	3	4	5	6	7
Index *x*	12	4	18	18	0	6	4
Key *d*	638	338	1104	1172	306	594	548

d. Generate ciphertext using the equation $E(x) = (7x + d) \bmod 36$. The plaintext MESSAGE with the seed 12 is encrypted as CGGCSYA as shown in Table 8.

Table 8: Equivalent ciphertext

Plaintext	M	E	S	S	A	G	E
Position *c*	1	2	3	4	5	6	7
Index *x*	12	4	18	18	0	6	4
Key *d*	638	338	1104	1172	306	594	548
$E(x) = (7x + d) \bmod 36$	2	6	6	2	18	24	0
Ciphertext	C	G	G	C	S	Y	A

To perform decryption using the modified Affine cipher, the following detailed processes are executed:

a. Identify ciphertext value and unique keystream. For instance, the ciphertext is CGGCSYA and the unique keystream is 980 596 1542 1622 552 924 866.

b. Generate plaintext using the equation D(y) = 31 (y - d) mod 36. The plaintext CGGCSYA with the keystream is 638 338 1104 1172 306 594 548 is decrypted as CGGCSYA as shown in Table 9.

Table 9: Equivalent plaintext

Ciphertext	C	G	G	C	S	Y	A
Key *d*	638	338	1104	1172	306	594	548
$D(y) = 31 (y - d) \bmod 36$	12	4	18	18	0	6	4
Plaintext	M	E	S	S	A	G	E

3. RESULTS AND DISCUSSION

In order to assess the viability of the proposed method, it is tested using a variety of plaintext and keys, then compared against the standard Affine cipher. The following test cases are shown in Tables 10-14.

Table 10: Test case 1

Plaintext	UNNEEDED
Plaintext Size	8 bytes
Seed (for modified method only)	14
Standard Affine Ciphertext	EVVCCXCX
Modified Affine Ciphertext	MMA0A40U

Table 11: Test case 2

Plaintext	BBBBBBBBBBBBBBB
Plaintext Size	15 bytes
Seed (for modified method only)	24
Standard Affine Ciphertext	NNNNNNNNNNNNNNN
Modified Affine Ciphertext	YG0MA0SMIGGIMS0

Table 12: Test case 3

Plaintext	AABBAABBAABBAABBAABBAABB AABBAABBAABB
Plaintext Size	36 bytes
Seed (for modified method only)	3
Standard Affine Ciphertext	IINNIINNIINNIINNIINNIINNIINNIINN IINN
Modified Affine Ciphertext	U40GAS4SMC40US04Y4S4YCG0UG 0SMGGGACS0

Table 13: Test case 4

Plaintext	CRYPTOGRAPHYISMAGIC
Plaintext Size	19 bytes
Seed (for modified method only)	10
Standard Affine Ciphertext	SPYFZAMPIFRYWUQIMWS
Modified Affine Ciphertext	OACYIGM4US0C6GCSGM6

Table 14: Test case 5

Plaintext	AFFINECIPHERENCRYPTIONAND DECRYPTIONPROCESS
Plaintext Size	42 bytes
Seed (for modified method only)	8
Standard Affine Ciphertext	IHHWVCSWFRCPCVSPYFZWAVIVX XCSPYFZWAVFPASCUU
Modified Affine Ciphertext	SSOAYG4A4IA60GOYCMIM60UYSU ASMSASOA44MAOIUM

Results show that the modified method performs better in producing ciphertext, especially for plaintext values with identical characters. As seen in Tables 11-12, characters A and B are encrypted without obvious patterns using the modified method as opposed to the ciphertext generated by the standard method. Cryptanalysis is made more difficult and challenging since the modified Affine cipher uses dynamically generated keystream using a random seed value. These findings make the proposed method more secure and complex compared to traditional substitution ciphers.

4. CONCLUSION

In this paper, a random seed value is added to the Affine cipher to generate a unique keystream that enables the cipher to produce unique ciphertext characters despite having identical letters from the plaintext. The vulnerability issue of the standard Affine cipher that is rooted in the plaintext structure is addressed by increasing its complexity. The enhanced Affine cipher generates result with distinct substitution values enabling a more secure encryption and decryption process.

REFERENCES

[1] 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.

[2] 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

[3] A. P. U. Siahaan, “Three-Pass Protocol Concept in Hill Cipher Encryption Technique,” *Semin. Nas. Apl. Teknol. Inf.*, pp. 31–35, 2016. https://doi.org/10.31227/osf.io/ek943

[4] M. Maxrizal and B. D. Aniska Prayanti, “Application of Rectangular Matrices: Affine Cipher Using Asymmetric Keys,” *CAUCHY –Jurnal Mat. Murni dan Apl.*, vol. 5, no. 4, pp. 181–185, 2019. https://doi.org/10.18860/ca.v5i4.4408

[5] T. M. Aung and N. N. Hla, “A Complex Polyalphabetic Cipher Technique Myanmar Polyalphabetic Cipher,” in *2019 International Conference on Computer Communication and Informatics, ICCCI 2019*, 2019, pp. 1–9. https://doi.org/10.1109/ICCCI.2019.8821797

[6] O. Laia, E. M. Zamzami, Sutarman, F. G. N. Larosa, and A. Gea, “Application of Linear Congruent Generator in Affine Cipher Algorithm to Produce Dynamic Encryption,” *J. Phys. Conf. Ser.*, vol. 1361, no. 1, pp. 1–6, 2019.

[7] A. G. Konheim, *Computer Security and Cryptography*. Wiley, 2007. https://doi.org/10.1002/0470083980

[8] H. Zhu, C. Zhao, X. Zhang, and L. Yang, “An image encryption scheme using generalized Arnold map and affine cipher,” *Opt. - Int. J. Light Electron Opt.*, vol. 125, pp. 6672–6677, 2014. https://doi.org/10.1016/j.ijleo.2014.06.149

[9] Y. S. Mezaal and S. F. Abdulkareem, “Affine Cipher Cryptanalysis Using Genetic Algorithms,” *JP J. Algebr. Number Theory Appl.*, vol. 39, no. 5, pp. 785–802, 2017. https://doi.org/10.17654/NT039050785

[10] S. A. Babu, “Modification Affine Ciphers Algorithm for Cryptography Password,” *Int. J. Res. Sci. Eng.*, vol. 3, no. 2, pp. 346–351, 2017.