International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 06 | June - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

XorShift And Random Number Generator for Image Encryption

Dr. Ayesha Ameen[1], Syed Danish Muazzam[2], Syed Abdul Mukthadher Mohiuddin[3], Talha Baig[4]

[1] Head of the Department, Department of Information Technology, Deccan College of Engineering and Technology.
[2,3,4]UG, Department of Information Technology, Deccan College of Engineering and Technology.

***Abstract*– With the rapid development of computer science, fast and secure transmission of data has gained great importance. Undoubtedly, one of the most common data types is digital images. The attractiveness of digital images is the result of wide range usage from social media to the defense industry. In order to transmit through correct and secure channels, images must be encrypted before they are sent. The relevant process is realized by means of encryption algorithms. Symmetric stream encryption algorithms which have been proposed so far have weaknesses in terms of speed and processing power. Therefore, encryption quality drops dramatically in certain scenarios. A novel symmetric stream encryption algorithm called XorShiftAnd has been proposed in this study.**

I.INTRODUCTION

Image encryption involves transforming a digital image to make it unrecognizable. Encryption methods are classified into symmetric and asymmetric algorithms. Asymmetric encryption uses public and private keys, while symmetric encryption relies on a single key and is preferred for its speed and simplicity. Symmetric encryption is further divided into block ciphers and stream ciphers. Block ciphers encrypt data in fixed-size blocks, whereas stream ciphers generate a random key stream using pseudo-random number generators (PRNGs). PRNGs, based on a seed value, work deterministically and have weaker randomness than true random generators.

Studies like that of Çakır examined the working principles of PRNGs and LFSRs, identifying key parameters for efficiency. Statistical tests have been used to assess the reliability of pseudo-random key streams. Various image encryption methods have been analyzed, including digital signatures, chaotic systems, and reflow techniques. Atalay et al. compared encryption algorithms, highlighting that Vigenère is weak against frequency attacks, DES is vulnerable to differential attacks, AES is costly, and RC4 is fast with good statistical performance. The BZ algorithm, especially when combined with chaos theory, has shown promising results in steganography. Zhu et al. proposed an encryption method involving bitwise permutation and pixel diffusion using Arnold maps. Dogan and Celik developed a hybrid algorithm using affine and Caesar ciphers with a zigzag scanning model for pre-encryption. Rad et al. utilized scan models and XOR functions to prevent data loss. A new stream cipher called XorShiftAnd was developed using a PRNG-based XOR shift, and successfully tested on grayscale images.

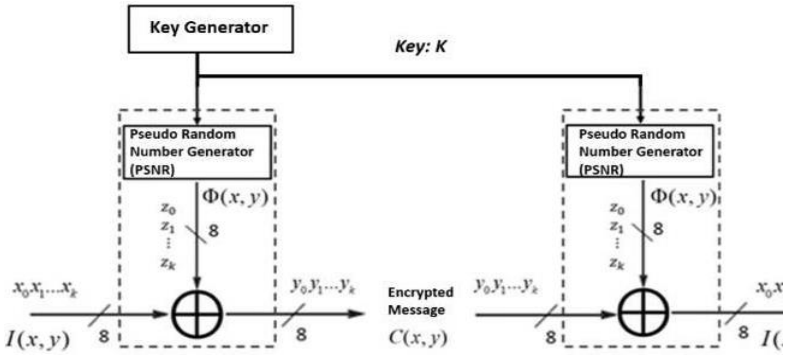
II. SYMMETRIC ENCRYPTION ALGORITHM

With the development of computer science and technology, there have been changes in encryption techniques. However, distribution and protection of keys is still a current problem to be solved. Symmetric algorithms are relatively affected by related negative features. This is because the encryption key is also used in the decryption process. Therefore, it is necessary to increase security measures by complicating existing algorithms.

Image encryption is defined as making original image unrecognizable with mathematical modifications on pixel data whereas decryption process is converting encrypted image to its original form. Typical stream encryption process of images is shown in Figure 1. The key, *K* which is produced by the key generator is employed to create a random number sequence, *zk*. It is also called key stream. Therefore, this process is called stream encryption. The approach algorithm is symmetric encryption since *K is* used both in encryption and decryption phases. Starting with the pixel in the upper left corner of the digital image, *xk* represents k^{th} pixel while *zk* represents the number which is produced by PRNG for the same pixel position and *yk* represents the encrypted pixel for the corresponding position. Image encryption process is typically defined as

$$C(x, y) = I(x, y) \oplus \Phi(x, y)$$

$$I(x, y) = C(x, y) \oplus \Phi(x, y)$$



where *I(x,y)* represents the original image, *Φ(x,y)* is a random key stream matrix and *C(x,y)* is an encrypted image. Furthermore, corresponds to the logical Xor operator. Performance of PRNG and length of K vector are the most critical features for stream cipher. The *K* vector, which contains seed values of PRNG, is produced by a key generator. Period and entropy of created key sequence concerns with predictability of the algorithm. Therefore, the longer key vector *K* means greater security against brute force attacks.

III. RANDOM NUMBER GENERATOR

Pseudo random number generators are small algorithms that use mathematical formulas to produce random numbers. Stream encryption algorithms require pseudo random number generators to make key streams. A typical pseudo number generator takes an initial value as input called seed and then number sequence is generated.

Random numbers could be reproduced if the seed is known. Therefore, the seed data is decisive and important. A typical example of PRNG, linear congruential generator (LCG) proposed by Thomson and Rotenberg in 1958 is defined as

$$zk + 1 = (a\, zk + c) \bmod m$$

where z0 is seed value and a, c as constant parameters. With the variable, m, the maximum value of the generated number is determined. To generate numbers for image encryption, m must be 255 or 8 bits. On the other hand, it is also possible to generate numbers in the range of [0, 1] with chaotic maps. For instance, logistic map is defined as

$$zk+1 = \mu$$

$$zk\,(1-zk\,)$$

where μ is known as the growth rate parameter. By choosing a suitable seed, z0 as initiation parameter, it is possible to generate a number sequence with long-periods.

Another number generator that is frequently used today and inspired by the logical Xor operator is Xorshift algorithm. The Xorshift random number generator, also called the shift register generator, is a pseudo random number generator suggested by Marsagli. Xorshift generates number sequences in two ways by shifting right or shifting left. Right shifting version is defined as

$$zk+1 = zk \oplus (zk >> s)$$

where *zk* represents the state of the generator and *s* represents shifting value. *z0* will be seed data of PRNG. As stated in equation (4), *zk* is shifted into right by *s* bits, a new number is generated by Xoring itself. The initial parameter must be properly chosen to obtain a sequence with a long period.

The criteria proposed by the German federal office for information security to evaluate random number generators are typical. It is emphasized that produced random number sequences should be different from each other. Also, the numbers that have been produced before or that will be produced in the future could not be accessed through any number produced within the scope of cyber security.

Another example of measure used to evaluate a random number generator is entropy which is defined as a measure of unpredictability of produced number sequences by PRNG. It is defined as

$$H = -\sum_{i=1}^{m=1} pi \ln pi$$

where *pi* represents the probability of i^{th} number generated. Greater value of entropy means greater disorder of data. For example, in an ideal generator which produces random numbers between [0, 255], the probability of each value should be *1/256 (0.00390)*. Consequently, entropy of an ideal distribution of random numbers generated by a PRNG should be *5.5448* with natural logarithm.

IV. XORSHIFT AND ALGORITHM

Xorshift approach which was stated in the equation, produces very large numbers in a very short time due to bit shifting and returns to the seed value again. On the other hand, 8-bits number is assigned to pixels in each channel of digital images. Thus, the number sequence must be in the range of [0,255]. Also, bit selection problems arise due to generated large numbers. To prevent Xorshift's negative features and benefit from the speed factor for symmetric encryption, a new algorithm called XorShiftAnd192 has been proposed as follows

$$Z = (z \oplus (z >> s) + \square a\ z\ \square)\ AND$$
$$0 \leq a < 1$$

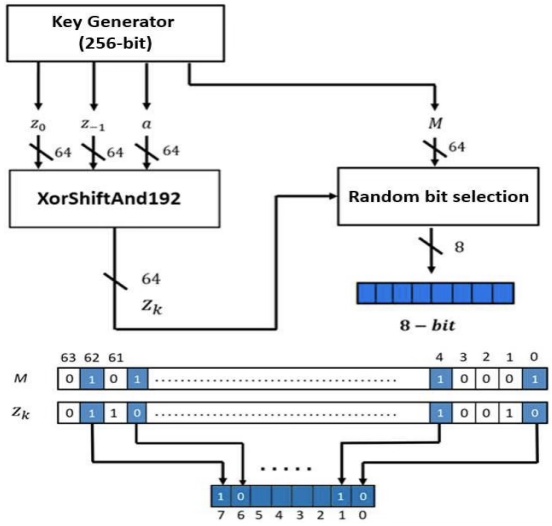
where *zk* and *zk-1* are states of PRNG. *s* represents shifting value. As could be seen, apart from the Xorshift algorithm, *a* and *zk-1* parameters were added. The parameter, *a, is a* decimal number that was inserted to disrupt

the periodicity of the Xorshift algorithm. Furthermore, the number of states is increased with zk and *zk-1* . Thus, one step and two step delayed state variables help to increase complexity or predictability of generated number sequences. Additionally, multiplication of *a* and *zk-1* is processed by the flooring function. Subsequently, predictability of the generated number sequence is further increased. Moreover, any *q* bit number sequence could be produced by means of the AND operator given in the equation.

For example, when the parameter, *q* is selected as 64 bit, a 64-bit integer number sequence is produced. On the other hand, the numbers produced for image processing must be 8- bits. In this study, a random bit selection approach has been proposed as shown in Figure 2. Only 8-bits from a 64-bit number is randomly chosen by means of bit mask, *M*. In fact, the parameter *M* is a component of the key vector. The critical property of *M* is that it must contain only ones, 1s with a number of 8. The bits across the every 1s in *zk* are used to construct 8 bit integer numbers. Consequently, randomness of the key stream was also increased as shown in Figure2. Finally, the key vector which is produced by key generator is define as

$$K = (z_0, z_{-1} , a, M)$$

where seed information and key mask are converted to binary. Consequently, a new Pseudo random number generators called XorShidtAnd256 has been achieved and tested with gray scale images.



V. EXPERIMENTAL RESULTS AND DISCUSSION

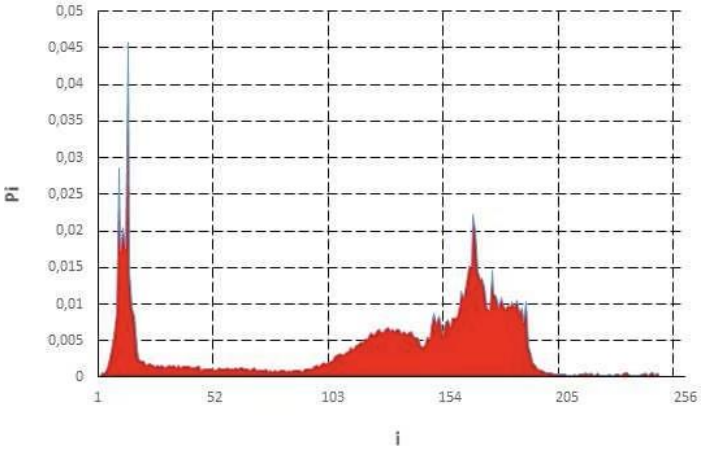
Pseudo-random number generators are algorithms that generate sequences of numbers starting from specified initial conditions. In this study, a random number generator based on the structure of the Xorshift number generator has been developed for image encryption. Although the Xorshift approach has a high rate of speed in terms of encrypting images, it has negative aspects such as low period and repetition. Therefore, a new algorithm called XorShiftAnd256 has been developed and tested with a user interface designed with C# environment. In the experimental study, the Cameraman image shown in Figure (a) was used. Histogram is a distribution of pixels in an image. Typically, i^{th} probability of gray level image is defined as

$$p = \frac{hi}{} , \qquad 0 \le i \le L$$

MxN

where *MxN* represents dimension of image while *hi* is probability of i^{th} intensity level and *L* represents maximum intensity level. Histogram of original Cameraman image is shown in Figure (b).

(a)



(b)

In order to evaluate performance of the image encryption process, similarity measurements between encrypted image and original image should be implemented. One of typically used metrics is mean squared error(MSE) which is defined as

$$MSE = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [I1(x,y) - I2(x,y)]$$

where *I1* and *I2* represent the compared images, while x and y represent pixel positions. On the other hand, the peak signal noise ratio (PSNR) is commonly used comparison metric stated as

$PSNR = 10\log(MSE)$

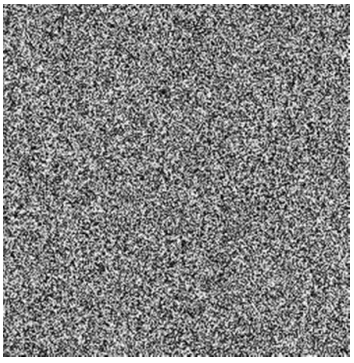
The pearson correlation coefficient (PCC) is a frequently preferred measurement tool for comparing images. Mathematically, the PCC is defined as follows

$$PCC = \frac{i \sum (xi - \mu x)(yi - \mu)}{}$$

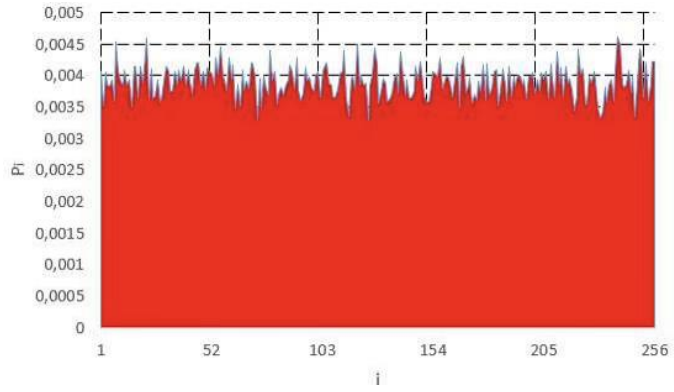
where *xi* represents the first image's intensity while *yi* represents the second image's intensity. *μx* represents the first image's average while *μy* represents the mean of the second one. High values of PCC means that the related images are quite similar. Therefore, the PCC in the image encryption process should be as low as possible.

Image encryption process was defined in Figure1 where *Φ(x,y)* is a random key stream matrix and *C(x,y*) is an encrypted image. As a 256x256 sized Cameraman image was used in the experiment, the size of the random number generated must be equal to size of image. Accordingly, random numbers generated with XorShidtAnd256 are visualized as shown in Figure 4(a) whereas its probability distribution is given in Figure 4(b). The components of key vector assigned are *z0*: (24006b91d495fa00)16, *z-1*:(3b38d61f11a25400)16, *a*: 0,81787566346257 and *M*: (043080020004c000)16. As
could be seen in Figure 4 (b), the probability distribution of random numbers is quite homogenous. Also the entropy of the corresponding distribution, *H= 5.5432* is very close to the ideal entropy value which is *5.5448* from equation.

When the random numbers or key stream shown in Figure 4(a), *Φ(x,y)* are Xored with the original Cameraman image given in Figure 3(a), encrypted image C*(x,y*) has been obtained as shown in Figure 5(a). As could be seen in Figure 5 (a), the encrypted image is quite unrecognizable. The *PCC* between encrypted and original Cameraman is *0.0036438.* It is quite low and confirms the visual complexity of encrypted images. Additionally, entropy of encrypted Cameraman image is calculated as *H: 5.5435* which is also very close to ideal distribution *5.5448.*

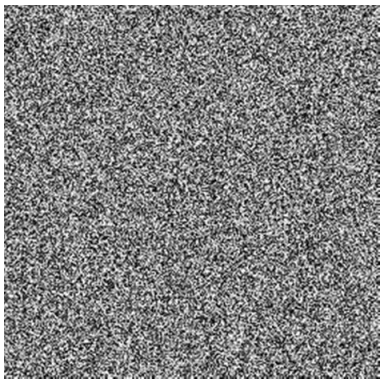


(a)

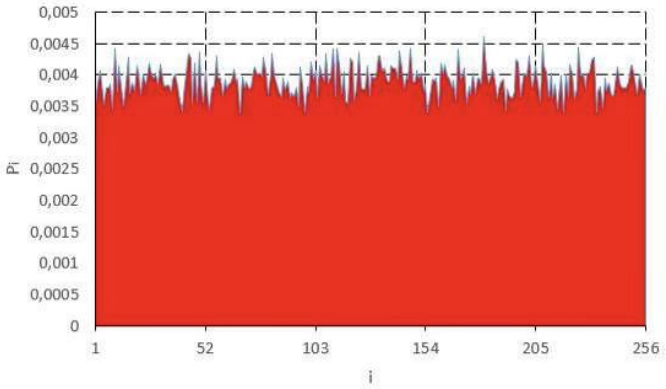


(b)

Fig.4.Output of Xorshift 256, *z0:* (24006b91d495fa00)16,
z-1: (3b38d61f11a25400), *a*: 0,81787566346257, *M:* (043080020004c000)
a) *Φ (x,y)* b) histogram, *H:* 5,5432



(a)

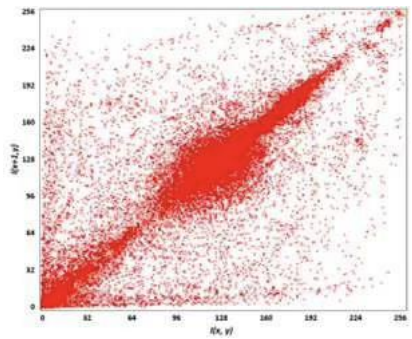


(b)

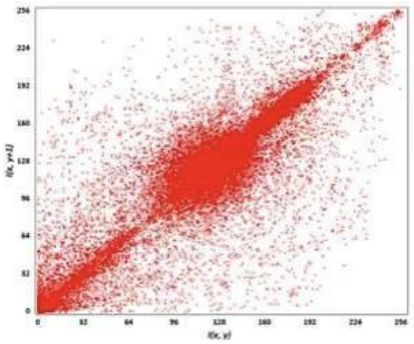
Fig. 5. *C(x,y):*Encrypted Cameraman image with Xorshift 256,
z0: (24006b91d495fa00)16, *z-1:* (3b38d61f11a25400)16, *a*: 0,81787566346257,
M: (043080020004c000)16 a) Encrypted , *PCC* = 0,0036438 b) histogram

Another information employed in performance comparison is scatter plot which is used to determine if there are patterns or correlations between two variables. In image encryption, related variables are corresponding to neighboring pixels in the image. Horizontal, vertical, and diagonal scatter plots of original Cameraman image are shown in Figure 6. There are almost linear correlations in each direction.

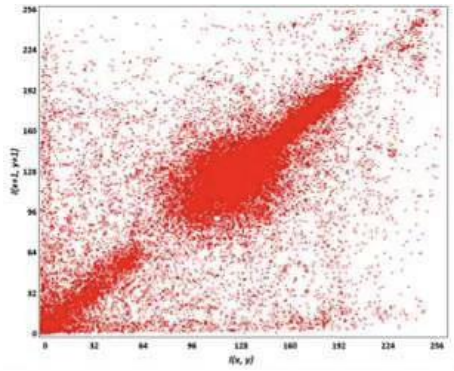
Scatter plots of encrypted Cameraman were given for three different directions in Figure 7. As can be seen from Figure 7 (a), Figure 7 (b) and Figure 7 (c), there was no accumulation in graphs. Besides, it was observed that points were homogeneously distributed. Therefore, it can be said that the adjacent pixels in the image are not correlated in three directions. Consequently, it confirms that the random number generator suggested, XorShiftAnd256 could be employed for image encryption applications.



(a)

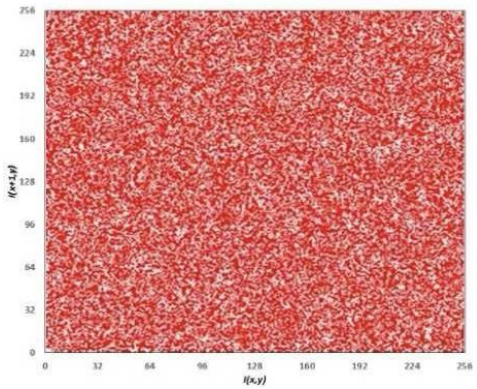


(b)

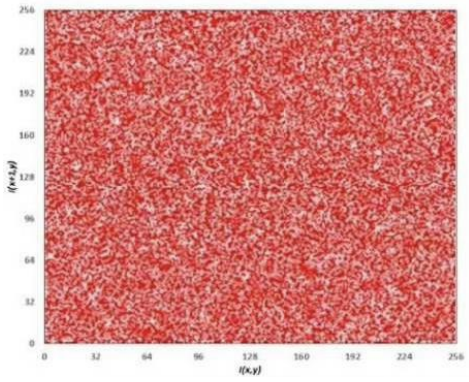


(c)

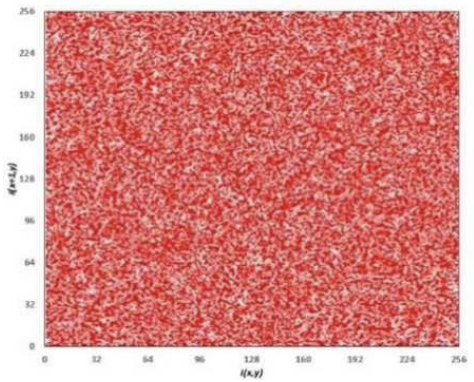
Fig. 6. Scatter plots a) horizontal b) vertical c) diagonal



(a)



(b)



(c)

Fig. 7. Encrypted Cameraman scatter plots: Xorshift 256
a) horizontal b) vertical c) diagonal

VI. CONCLUSION

Within the scope of this study, a new pseudo random number generator and stream encryption algorithm called XorShiftAnd256 have been proposed. The Xorshift algorithm was improved by adding a new state variable and an extra parameter and flooring function. Subsequently, the distribution of the random number sequence is homogenized. Furthermore, bit length of random numbers to be generated could be controlled with a logical AND operator. Also, the image encryption performance of the proposed algorithm was confirmed with a 256-bit key.

REFERENCES

1. K. Yıldırım ve H. E. Demiray , "Simetrik ve asimetrik şifreleme yöntemlerine metotlar: çırpılmış ve birleşik akm-vkm", *Gazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, c. 23, sayı. 3, ss. 0, Şub. 2013
2. A. W. Dent, “Hybrid Cryptography,” Cryptology ePrint Archive, Paper 2004/210, 2004. [Online]. Available: https://eprint.iacr.org/2004/210
3. F. Şahin, "Modern blok şifreleme algoritmaları," İstanbul Aydın Üniversitesi Dergisi, no. 17, pp. 47-60, (2015).
4. A. Beşkirli , D. Özdemir ve M. Beşkirli , "Şifreleme Yöntemleri ve RSA Algoritması Üzerine Bir İnceleme," *Avrupa Bilim ve Teknoloji Dergisi*, ss. 284-291, Eki. 2019, doi:10.31590/ejosat.638090
5. T. Etem ve T. Kaya , "Görüntü Şifreleme için Trivium-Doğrusal Eşlenik Üreteci Tabanlı Bit Üretimi," *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, c. 32, sayı. 1, ss. 287-294, Mar. 2020, doi:10.35234/fumbd.687403
6. B. Çakır, “Akış şifreleme algoritmalarının karşılaştırmalı olarak incelenmesi,” Yüksek Lisans Tezi, Elektrik Elektronik Mühendisliği, Hacettepe Üniversitesi, 2012.
7. E. Erkek, “Akış şifreleme algoritmaları kullanılarak rasgele sayı üretilmesi ve FPGA ortamında gerçekleştirilmesi,” Yüksek Lisans Tezi,Bilgisayar Mühendisliği, Fırat Üniversitesi, 2015.
8. H. Sakal ve M. Yildirim, “Görüntü şifreleme için scan paternlerini kullanan hibrit bir yöntem,” *Selçuk Teknik Dergisi*, 15(3), 264-283, 2016.
9. L. Bao and Y. Zhou, “Image encryption: Generating visually meaningful encrypted images,” Information Sciences, 324, 197-207, 2015. doi:10.1016/j.ins.2015.06.049

10. N. S. Atalay , Ş. Doğan , T. Tuncer ve E. Akbal , "İmge Şifreleme Yöntem ve Algoritmaları," *Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi*, c. 10, sayı. 3, ss. 815-831, Eyl. 2019, doi:10.24012/dumf.478877
11. Q. H. Makki, A. M. Abdalla and A. A. Tamimi, "A Survey of Image Encryption Algorithms," *2021 International Conference on Information Technology (ICIT)*, 2021, pp. 598-602, doi: 10.1109/ICIT52682.2021.9491727.
12. H. Zhu, C. Zhao, X. Zhang, and L. Yang, "An image encryption scheme using generalized Arnold map and affine cipher," Optik, vol. 125, pp. 6672–6677, 2014. doi:10.1016/j.ijleo.2014.06.149
13. N. Doğan ve H. Çelik , "Tarama Modeli Kullanan Karma Bir Görüntü Şifreleme Yöntemi," *Politeknik Dergisi*, ss. 1-1, Ara. 2022, doi:10.2339/politeknik.902661
14. R. M. Rad, A. Attar, and R. E. Atani, "A New Fast and Simple Image Encryption Algorithm Using Scan Patterns and XOR," International Journal of Signal Processing, Image Processing and Pattern Recognition, vol. 6, pp. 275–290, 2013. doi: 10.14257/ijsip.2013.6.5.25
15. S. Chandra, S. Bhattacharyya, S. Paira and S. S. Alam, "A study and analysis on symmetric cryptography," *2014 International Conference on Science Engineering and Management Research (ICSEMR)*, pp. 1- 8, 2014. doi: 10.1109/ICSEMR.2014.7043664.
16. S. K. Park and K. W. Miller, "Random Number Generators: Good Ones Are Hard to Find," Commun. ACM, vol. 31, no. 10, pp. 1192–1201, Oct. 1988, doi: 10.1145/63039.63042
17. W. E. Thomson, "A Modified Congruence Method of Generating Pseudo-random Numbers," Comput. J., vol. 1, p. 83, 1958.
18. S. C. Phatak and S. S. Rao, "Logistic map: A possible random-number generator," Phys. Rev. E, vol. 51, no. 4, pp. 3670–3678, Apr. 1995, doi: 10.1103/PhysRevE.51.3670.
19. G. Marsaglia, "Xorshift RNGs", *J. Stat. Soft.*, vol. 8, no. 14, pp. 1–6, Jul. 2003. doi: 10.18637/jss.v011.i05
20. W.Alexi, B. Chor, O.Goldreich, and C.P. Schnorr, "RSA and Rabin functions: Certain parts are as hard as the whole," SIAM Journal on Computing, vol. 17, no. 2, pp 194-209, 1988. doi: 10.1137/0217013.