Published Online First: July, 2024
https://doi.org/10.21123/bsj.2024.9280
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Proposed Color Image Lightweight Encryption using SALSA20 with Key Derivation Function

***Ebtehal Talib*[1], Abeer Salim Jamil*[2], Nidaa Flaih Hassan*[3]**

[1]Ministry of Higher Education and Scientific Research, Baghdad, Iraq.
[2]Department of Computer Technics Engineering, Al-Mansour University College, Iraq.
[3]Department of Computer Science, University of Technology, Baghdad, Iraq.
*Corresponding Author.

Received 08/09/2023, Revised dd/mm/yyyy, Accepted 02/01/2024, Published Online First 20/07/2024

Abstract

Due to the extraordinary development in the production and exchange of multimedia across unprotected networks, there is a greater need to develop reliable secure, and effective cryptographic algorithms. This paper proposes a new lightweight encryption algorithm combining SALSA20 and password-based Key Derivation Function2 (PBKDF2) for the encryption of color images. Due to the PBKDF2 characteristics such as strengthening the security of the password by applying multiple rounds of a pseudorandom function (PRF) to generate the key, Salt usage adds additional randomness to the key derivation process, therefor it is used to enhance SALSA20. By offering a large key space, the recommended lightweight image encryption has demonstrated robustness against brute force attacks. Additionally, the suggested algorithm capable of protecting against statistical cracking, differential attack, and image security is good based on the histogram, correlation, NPCR, UACI, and Entropy criteria. Encryption time for test images is very short, ranging from one to one and a half seconds, so it is close to being run in real time.

Keywords: Light Encryption, SALSA20, Key Derivation Function, PBKDF2, Pseudorandom Function, SHA256, Attacks.

Introduction

Most ciphering techniques fall into one of two categories: The first is a block cipher, which describes how the cipher works by dividing each original piece of data into sequential blocks, each of which is then encrypted using the same key [1,2]. The second is a stream cipher, which describes how the cipher works by using the XOR function to combine the original data with the key random sequence to produce the cipher data[3,4]. The encryption and decryption procedures are necessary to construct the identical keystream series using the same seed key [5, 6,7]. Modifying the key stream series prevents anyone from giving the adversary instructions on decrypting the data by ensuring that the keystream cannot be repeated [8, 9]. Stream cipher's advantages compared with block cipher include the lack of error propagation and reduced complexity [4,10,11]; additionally, it is designed to process more quickly [12,13], this paper used SALSA20 stream cipher for color image encryption.

Password-based Key Derivation Functions (PBKDFs) amalgamate user-provided passwords with a randomly generated salt and an iteration count to produce cryptographic keys for encryption systems[14, 15]. These functions are designed to streamline the handling of substantial cryptographic keys, ultimately improving user key management. RFC 8018 [13] encompasses a series of guidelines

Published Online First: July, 2024
https://doi.org/10.21123/bsj.2024.9280
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

about PBKDF1 and PBKDF2, offering a comprehensive point of reference for these key derivation techniques[16].

NIST [13] and RFC 8018 [17] recommend using PBKDF2 for developing new applications. Furthermore, they suggest employing a randomly generated salt of at least 128 bits and a minimum iteration count of 1000 [16]. The recommendations and the specific implementation instructions described in these documents have led to the development of multiple libraries in various programming languages. As a result, numerous systems, such as Apple's IOS mobile operating system[18], Wi-Fi Protected Access [17], and Firefox Sync for client-side password stretching [18], have incorporated the PBKDF2 algorithm, which is also employed in the context of this study.

The organization of this paper is as follows: The SALSA20 structure is explained in Section 2. In Section 3, the PBKDF2 is defined. The description of the proposed image lightweight encryption steps is presented in Section 4, and the evaluation metrics with their equations are explained in Section 5. After that, the results and discussion are described in Section 6. Finally, the conclusion is clarified in section 7.

SALSA20 structure

The stream cipher SALSA20 employs a counter mode for encryption, beginning with 4× 4 arrays containing 512 bits [19, 20]as depicted in Table 1.

Table 1. SALSA20 Distribution Array

Constant 1	Key 1	Key 2	Key 3
Key 4	Constant 2	Nonce 1	Nonce 2
Counter 1	Counter 2	Constant 3	Key 5
Key 6	Key 7	Key 8	Constant 4
		=	
V0	V1	V2	V3
V4	V5	V6	V7
V8	V9	V10	V11
V12	V13	V14	V15

As shown in Fig. 1, the core operations within SALSA20 consist of 'addition, XOR, and rotation,' and they are applied iteratively on a SALSA [20] array over 10 rounds [19]. SALSA20 pertains to the array that undergoes two transformations during each round. After the SALSA20 process, the final alteration of the SALSA20 array is combined with the initial seed for the SALSA20 array using an addition operation [11].

Ninety-six-word operations have been completed in each round of SALSA20 consisting of forty-eight-word operations for the first change and forty-eight-word operations for the second change. Forty-eight-word operations are obtained by multiplying sixteen-word operations by three (addition, XOR, and rotation)[19]. Nine hundred and sixty-word operations are used in total throughout ten rounds. Nine hundred and sixty-word operations + sixteen-word operations bring the total word operations for one encryption to one hundred and seventy-two at the end of the SALSA20[21].

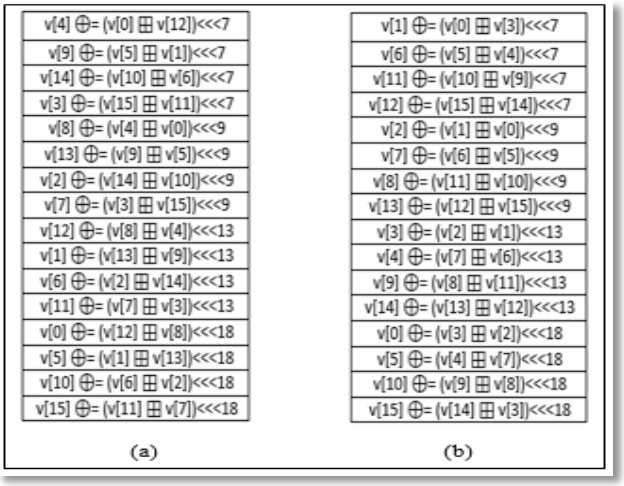

Figure 1. Operations of SALSA20: (a) first Changing. (b) Second Changing

Password-Based Key Derivation Function 2

A user-supplied password or passphrase transforms into a cryptographic key through the utilization of PBKDF2[17], which necessitates the inclusion of the following four input parameters: password,

- S: salt,
- C: number of iterations,
- K_{len}: Key length.

Subsequently, it produces the resulting key by employing a pseudo-random function (PRF) for 'C' iterations. Among the input parameters[17], an authorized user must keep the password confidential. A randomly generated salt value enhances the search space and deters dictionary attacks. The iteration count determines how often the underlying PRF is executed to create the key. This approach aims to introduce CPU-intensive tasks that hinder attackers from conducting exhaustive searches[18].

Published Online First: July, 2024
https://doi.org/10.21123/bsj.2024.9280
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

As the PRF, SHA (1, 224,256, 384,512, 512/224, and 512/256) are all supported by PBKDF2. The length of the resulting key in PBKDF2 is not fixed and can vary. However, its size is constrained by the hash function properties used as the Pseudo-Random Function (PRF)[17]. The PRF's output length is denoted as (h_{Len}), and it's crucial to ensure that the derived key's length (dk_{Len})remains within the limit of (2^{32} - 1) *h_{Len}. In our analysis within this research, we utilize the SHA-256 PRF. The SHA-256 algorithm is described in the paragraphs that follow[22]. Initially, the original message M is subjected to Merkle-Damgaard strengthening by padding it to a size that is a multiple of 512 bits. This padding process includes adding a solitary '1' bi, equalde 6equalilreffi is divided ito fr separate 52-bts ocks: M0, M1, and ML-1[23].

$$H_0 = IV, H_{l+1=} CF(M_l, H_l),\ 0 \le l < L \dots\dots\dots 1$$
18

where :

He is an 8-word, 256-bit chaining value; each word comprises 32 bits.

+ stands for addition in modulo 2^{32}

The result of the hash function is the last chaining value, H_L

The compression function comprises two primary elements: message expansion and the state update transformation denoted as Cf(Ml, Hl)[22.]

Expansion of Message:
The message scheduling process of SHA-256 divides Ml into 16 (32-bit) message words, labeled as m0 through m15. It then proceeds to expand these 16 words into 64 (32-bit) words, represented as Wi, where i ranges from 0 to 63[23], as described by the following equation:

as described by the following equation:

$$W_i \leftarrow \begin{cases} m_i & 0 \le i < 16 \\ \sigma_1(W_{i-2}) + W_{i-7} + \sigma_0(W_{i-15}) + W_{i-16} & 16 \le i < 63 \end{cases} \dots..2$$
23

definitions ofthe unctions $\sigma_0\ (x)$ and $\sigma_1\ (x)$ are as follows :

$$\sigma_0(x) \leftarrow (x\ >>> 7) \oplus (x \gg 18)\ \oplus (x \gg 3) \dots 3$$
23
$$\sigma_1(x) \leftarrow (x\ >>> 17) \oplus (x \gg 19)\ \oplus (x \gg 10) \dots 4$$

Where:

$\oplus$ Mean XOR operation

$>>>$ And $>>$ present left rotation and left shift, respectively [19].

Updating The State Transformation :

The step function is iteratively applied 64 times to update the 256-bit chaining value H_l. Let's denote $pi = ai \parallel bi \parallel ci \parallel di \parallel ei \parallel fi \parallel gi \parallel hi$ as the input for step i (where i = 0, ⋯, 63). In other words, $p0$ is initialized as Hl, and $p64$ represents the output value of step 63. The transformation process for each step is illustrated in Fig.2 [22.] It's important to note that Ki denotes the round constant, while the functions Ti, Ch, Maj, ∑0, and ∑1 are defined as follows [22]:

$$T_i = h_i + \sum_1 (e_i) + Ch(e_i, f_i, g_i) + K_i + W_i \dots\dots\dots 5$$
$$Ch(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z) \dots\dots\dots 6$$
$$Maj(x,y,z) = (x \wedge y) \oplus (y \wedge z) \oplus (x \wedge z) \dots\dots\dots 7$$
$$\sum_0 (x) = (x\ >>> 2) \oplus (x \gg 13)\ \oplus (x \gg 22) \dots 8$$
$$\sum_1 (x) = (x\ >>> 6) \oplus (x \gg 11)\ \oplus (x \gg 25) \dots 9$$

$H_{l+1} = p_{64} + H_l$ is set as the chaining value for the subsequent message block M_{l+1} after processing 64 expanded message words W_i [19].

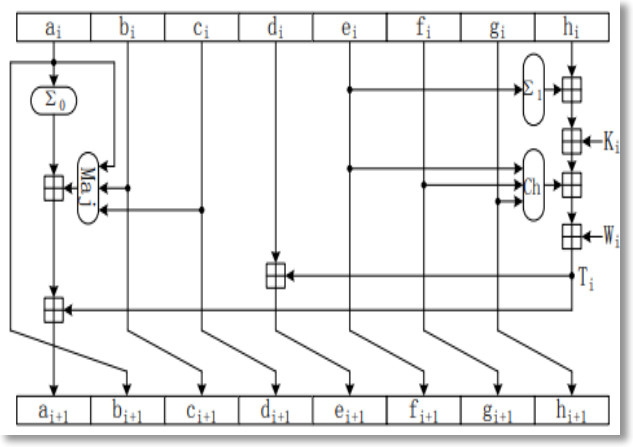

Figure 2. SHA-256's step function

Published Online First: July, 2024
https://doi.org/10.21123/bsj.2024.9280
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Methods

Related Work

Different papers were used to develop SLASA20. The super SALSA keystream suggested in [21] makes use of different volume matrices (array (4, 4), array (4, 8), and array (4, 16)) to increase the complexity of the key stream and make it more resistant to differential and linear attacks. Additionally, because the volume of one array element is not constant, the effect of modifying the volume will cause the diffusion of the generated keystream to rise with each iteration. To have the proper combination and more robust security, [19] suggests using the approach of passing the Gost keys to the SALSA stream. Due to its inconvenient technique, a brute force attack should not be employed in this circumstance as it requires 2256 likely keys to break keys. A new SALSA20 variation that takes advantage of chaos theory and can accomplish diffusion more quickly than the original SALSA20 is introduced in[4]. A battery of tests has been conducted to evaluate and benchmark the approach against the original SALSA20. The majority of testing demonstrates that, while offering the same diffusion level, the suggested chaotic SALSA of two rounds is faster than the original SALSA20/4's four cycles. in [22] The SALSA20 family of reduced-round ciphers, which includes the (8,12) round cipher, is recommended for users who prioritize speed over assurance. It makes use of a hash algorithm and a 256-bit key. A successful fusion improves the weaknesses of the SALSA20 algorithm by boosting its unpredictability by utilizing both the advantages of the random maps and the SALSA20 algorithm. To create a strong keystream that is sufficiently random to evade adversaries' predictions, accomplish good diffusion, and withstand known attacks.

The Proposed Color Image Lightweight Encryption based on PBKDF2

The proposed new color image lightweight encryption methodology is broken down into two parts, as shown in Fig.3. The first part of the suggested process generates a strong derived key using PBKDF2. The second part of the recommended process uses the SALSA20 structure for Image encryption based on the derived key to improve efficiency. Algorithm (1) explains the steps of the proposed encryption algorithm. In the proposed algorithm, SALSA20 was chosen because it is a lightweight algorithm and its work was improved by preparing a strong key based on a password-based key derivation function. PBKDF2 depends on the password, Salt is a set of random bytes, which adds a kind of randomness to it in addition, it depends on the Pseudorandom Function (PRF), itis repeated 1000 times, and here SHA-256 was used as PRF. This derived strong key was used with Nonce to encrypt the color image using SALSA20.

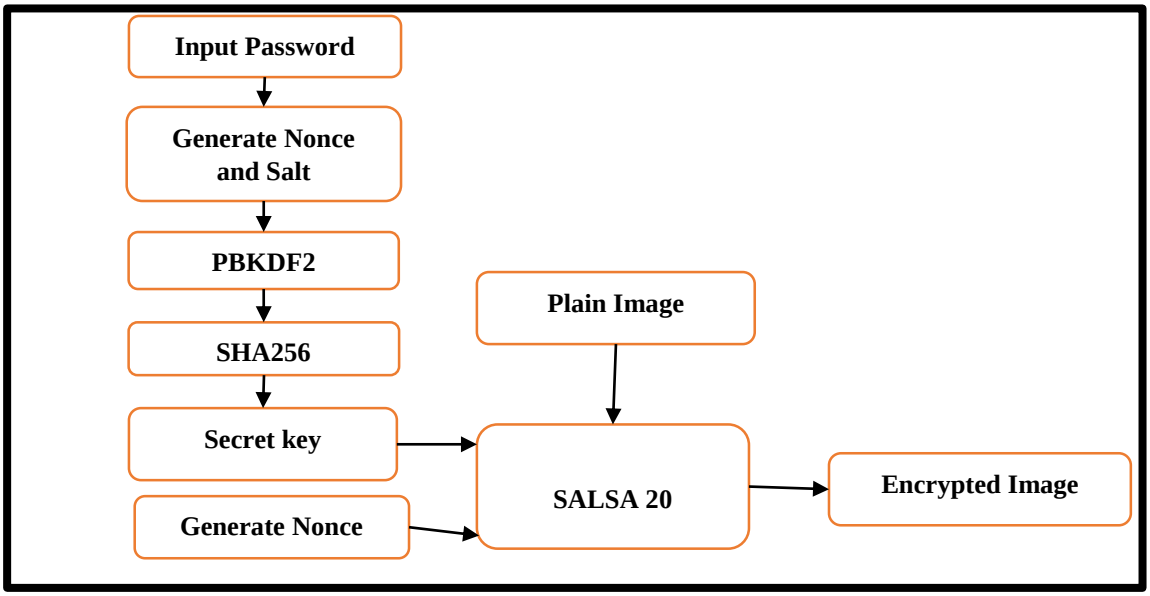

Figure 3. Diagram of Proposed Method

Published Online First: July, 2024
https://doi.org/10.21123/bsj.2024.9280
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Algorithm (1): The Proposed Encryption Algorithm
Input: Plain color image, password
Output: Encrypted image
Step1: Read the color image
Step2: Obtain the width and height of the image (W, H)
Step3: Generate a random nonce
Step4: Read the specified password
Step5: Generate a random salt
Step6: Generate a key by performing 100,000 iterations using PBKDF2 to derive a
The 256-bit encryption key and SHA256 as the hashing algorithm
Step7: For i = 0 to W -1
For j = 0 to H -1
Encrypts each pixel (i,j) using SALSA20 the derived key and nonce,
End for
End for
Step8: Return the encrypted image

Performance Evaluation Metrics
The computation of the various metrics used to assess the effectiveness of the suggested image encryption algorithm is provided in this section. Those metrics are the Number of Pixels Change Rate (NPCR), Unified Average Changing Index (UACI), Human Visual System and Histogram Analyses, Entropy Analysis, and Correlation Coefficient.

NPCR and UACI Analysis
The NPCR and UACI metrics assess the encryption algorithm's resistance to differential attacks. NPCR considers the pixel differences between the original and encrypted images, and its calculation is described by Eq. 10. A higher NPCR value indicates a greater level of pixel randomization[14].

$$[7].\text{NPCR} = \frac{\sum_{i,j} D(i,j)}{W*H} * 100\% \ldots\ldots 10 \quad [14]$$

UACI, as detailed in Eq. 11, gauges the average intensity of differences between the original image and its encrypted counterpart

$$\text{UACI} = \frac{1}{L}\left[\sum_{i,j} \frac{P(i,j) - C(i,j)}{255}\right] * 100\% \ldots\ldots 11 \quad [14]$$

D(i, j) is :

$$\text{D(i,j)} = \begin{cases} 0 & P(i,j) = C(i,j) \\ 1 & P(i,j) \neq C(i,j) \end{cases} \quad 12 \quad [14]$$

P(i, j) and C(i, j) denote the pixel values of the original and encrypted images, respectively. Additionally, W and H represent the image's width and height, while L represents the maximum pixel value found in both images[16].

Correlation Coefficient Analysis
Another crucial statistic for assessing an encryption algorithm's efficacy is the correlation coefficient analysis (r). Calculating the correlation coefficient between two pixels in the x and y yields a value that ranges between -1 and 1 [22]. The correlation coefficient between neighboring pixels should be near zero to imply no correlation for an encryption scheme to resist statistical attacks[14]. Otherwise, values close to 1 and -1 indicate strong positive and negative correlations, respectively, making it possible for statistical assaults to reveal an encrypted image. Eq 12 explains how the correlation coefficient is determined [16]:

$$r_{x,y} = \frac{\text{cov(x,y)}}{\sqrt{D(x)}\sqrt{D(y)}} \ldots\ldots\ldots 13 \quad [16]$$

$$\text{cov}(x,y) = \frac{1}{N}\sum_1^N (X_i - E(x))(y_i - E(y)) \ldots. \quad 14 \quad [16]$$

$$E(x) = \frac{1}{N}\sum_1^N X_i \ldots.. 15 \quad [16]$$

$$D(x) = \frac{1}{N}\sum_1^N (X_i - E(x))^2 \ldots.16 \quad [16]$$

X and Y represent the original and encrypted images and surrounding pixels. D(X) represents X's variance, while E(X) represents X's expected value. Cov (X, Y) represents the covariance between X and Y[22].

Human Visual System and Histogram Analyses
A human visual system (HVS) evaluation is the first and simplest metric for assessing image encryption technology. The image histogram is the second visual evaluation metric. An image's histogram

Published Online First: July, 2024
https://doi.org/10.21123/bsj.2024.9280
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

describes the probability density function for its pixels in the context of data encryption. A robust encryption method hides any statistical characteristics in the image by uniformly distributing the gray levels across the histogram[22].

Entropy

To assess an image based on the unpredictability of the distribution of its grey pixels, information entropy is utilized as a metric. Eq. 17 demonstrates how it is computed for a picture, where p(m_i) is the likelihood that any given symbol will appear among the image's total M symbols[15].

$$\mathbf{H(M)} = \sum_{i=1}^{m} p(m_i)\, log_2 \frac{1}{p(m_i)}) \ldots\ldots 17 \quad ^{14}$$

Results and Discussion

In this section, the results of the proposed algorithm are analyzed and discussed. The popular Lena image (512 × 512) in the image processing community is employed to test the performance of the proposed algorithm. The first column (a) in Fig. 4 displays the original image of Lena, the second column (b) represents the encrypted image, and the last column (c) represents the decrypted image.

Figure 4. (a) Plain Lena Image, (b) Encrypted Image, (c) Decrypted Image

Table 2 shows the performance evaluation values when the proposed algorithm is on the test image. The following set of criteria was used to evaluate the security analysis of the proposed algorithm.

Table2. Performance Evaluation of The Proposed Algorithm

Criteria	Encryption Image
The Entropy of the Original Image	13.46
Entropy	13.46
NPCR	99.62
UACI	50.06
Correlation Coefficient	-0.000778
Encryption time	1.3 seconds
Decryption time	1.3 seconds

Human Visual System: The palin and encrypted images are displayed in Fig.4. The encrypted image depicts illegible data.

Histogram Analyses: the proposed encryption algorithm is robust because it hides statistical characteristics in the image by uniformly distributing the gray levels across the histogram. This can be shown in Fig.5 and Fig.6, which demonstrate the difficulty of decrypting encrypted images and the suggested encryption algorithm's resistance to statistical attacks.

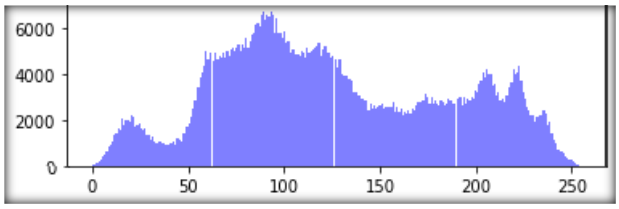

Figure 5. Histogram of Plain Image

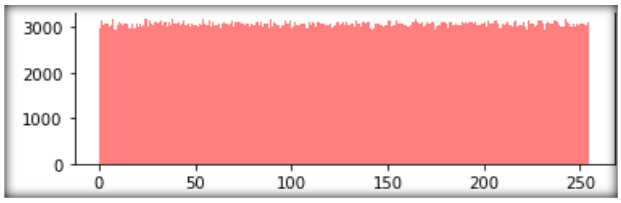

Figure 6. Histogram of Encrypted Image

Entropy: Entropy values for the original and encrypted images are 13.46 and 13.38, respectively. Given that both entropy values are similar, the encryption algorithm will likely successfully maintain the original image's randomness and information content. Although a small amount of entropy is expected to be lost throughout the encryption process, the technique retains much randomness.

Correlation coefficient: The correlation coefficient is -0.000778 between the original and encrypted

Published Online First: July, 2024
https://doi.org/10.21123/bsj.2024.9280
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

images, which means no linear relationship (or very weak) between them. Because there is no structure or apparent pattern that can be easily exploited to recover the plain image, this is considered desirable in encryption.

NPCR: Many pixels differ between the original and encrypted photos, as seen by the NPCR value of 99.62. A higher NPCR number often denotes an encryption technique with better randomness and diffusion.

UACI: The UACI value of 0.5004 indicates that, on average, there is a 49.97 difference in intensity between corresponding pixels in the original and encrypted images. A higher UACI value indicates that the encryption algorithm has more diffusion.

Encryption Time: The time to encrypt an image can vary depending on hardware capabilities and the image size. The encryption time for the proposed algorithm is 1.3 seconds, which means it can run on real-time programs.

The proposed algorithm's produced keys surpass the results of the five benchmark tests, as indicated in Table 3. The keystream is portrayed as having a high toughness and simple operations at random. The suggested approach demonstrated good performance when compared to related work and passed the test value criterion.

Table3. Five Standard Test

Criteria	Threshold	Proposed Algorithm	(1)	(2)	(3)	(4)	Test value< Threshold
Frequency	3.481	0.47	2.82	1.231	0. 62606	0.7667	pass
Serial	5.991	0.23	3.698	2.124	0.637119	0.8231	pass
Poker	24.995	10	8	14.912	12	-	pass
Run	12.591	3.18	4.61	7.852	6.4	8.867	pass
Auto-correlation	1.96	0.0078	0.336	0.814	0.0041727	-	pass

The proposed encryption algorithm shows promising characteristics based on these results. It can run in real-time programs, exhibits good diffusion properties, preserves the randomness and information content, and offers a low correlation between the original and encrypted images.

Conclusion

This paper proposes a new color image encryption algorithm by combining SALSA20 with PBKDF2. The proposal increases the security, safeguarding, confidentiality, and integrity of the encrypted data by utilizing PBKDF2 to derive an initial key for the SALSA20. SALSA20 used the strong key for the encryption color image. Five Standard Tests and different metrics, such as NPCR, UACI, entropy, histogram, and correlation coefficient analysis, were used to assess the performance of the proposed algorithm. The proposed algorithm showed good results, and it is resistant to different types of attacks in terms of execution time, the proposed algorithm is fast. For future work, apply the suggested algorithm to audio and video, and use different hash functions as a pseudorandom functions.

Authors' Declaration

- Conflicts of Interest: None.
- We hereby confirm that all the Figures and Tables in the manuscript are ours. Furthermore, any Figures and images, that are not ours, have been included with the necessary permission for re-publication, which is attached to the manuscript.
- Authors sign on ethical consideration's approval.
- Ethical Clearance: The project was approved by the local ethical committee at [1]Ministry of Higher Education and Scientific Research.
- No animal studies are present in the manuscript.
- No human studies are present in the manuscript
- No potentially identified images or data are present in the manuscript.

Published Online First: July, 2024
https://doi.org/10.21123/bsj.2024.9280
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

Authors' Contribution Statement

E.T. contributed to the design and drafting of the MS, A.S.J. contributed revision and proofreading and N. F. H. contributed analysis of the results.

References

1. Hassan NF, Abbas RJE, Journal T. Proposed Video Watermarking Algorithm based on Edge or Corner Regions. J Eng Technol 2018; 36(1 Part B): 25-32. https://doi.org/10.30684/etj.36.1B.4
2. Nassef M, Alkinani MH, Shafik AM. A Novel Image Cryptosystem Inspired by the Generation of Biological Protein Sequences. IEEE Access. 2023;11:29101-29115, https://doi.org/10.1109/ACCESS.2023.3261133 .
3. Lessage X, Collier L, Van Ouytsel CHB, Legay A, Mahmoudi S, Massonet P. Secure federated learning applied to medical imaging with fully homomorphic encryption. In: 2024 IEEE 3rd International Conference on AI in Cybersecurity (ICAIC). 2024. p. 1-12. https://doi.org/10.1109/ICAIC60265.2024.10433836 .
4. Mahdi MS, Hassan NF. Design of Keystream Generator utilizing Firefly Algorithm. JQCM, 2018; 10(3): 91-9. https://doi.org/10.29304/jqcm.2018.10.3.441
5. Salman D, Azeez R, Abdul-hossen A,. Build Cryptographic System from Multi-Biometrics using Meerkat Algorithm. IJCI ,.2019; 45(2): 1-8. https://doi.org/10.25195/ijci.v45i2.46
6. Sulaiman S, Hanapi ZM. Extensive analysis on images encryption using hybrid elliptic curve cryptosystem and hill cipher. J Comput Sci. 2021;17:221-230. https://doi.org/10.3844/jcssp.2021.221.230 .
7. Tayyeh HK, Mahdi MS, AL-Jumaili AA. Novel Steganography Scheme using Arabic Text Features in Holy Quran. Int J Electr Comput Eng. 2019; 9(3): 1910. https://doi.org/10.11591/ijece.v9i3.pp1910-1918
8. Salim KG, Al-alak SMK, Jawad MJ. Improved Image Security in Internet of Thing (IOT) Using Multiple Key AES. Baghdad Sci J. 2021; 18(2): 0417. https://doi.org/10.21123/bsj.2021.18.2.0417
9. Bersani F, Tschofenig H. The EAP-PSK protocol: A Pre-shared Key Extensible Authentication Protocol (EAP) method. 2007. RFC 4764. Report No.: 2070-1721. https://doi.org/10.17487/RFC4764
10. Kadir A, Hamdulla A, Guo W-QJO. Color Image Encryption using Skew Tent map and Hyper Chaotic System of 6th-order CNN. Optik . 2014; 125(5): 1671-5. https://doi.org/10.1016/j.ijleo.2013.09.040
11. Ali RH. Steganography in Audio Using Wavelet and DES. Baghdad Sci J. 2015; 12(2): 431-6. https://doi.org/10.21123/bsj.2015.12.2.431-436
12. Rimani R, Said NH, Pacha AALI, Ramos JAL. An Efficient Image Encryption Using a Dynamic, Nonlinear and Secret Diffusion Scheme. Baghdad Sci J. 2021; 18(3): 0628. https://doi.org/10.21123/bsj.2021.18.3.0628
13. Moriarty K, Kaliski B, Rusch A. Pkcs# 5: Password-based Cryptography Specification Version 2.1. RFC. 2017. Report No.: 2070-1721.
14. Kodwani G, Arora S, Atrey PK, editors. On Security of Key Derivation Functions in Password-based Cryptography. IEEE International Conference on Cyber Security and Resilience. 2021. https://doi.org/10.1109/CSR51186.2021.9527961
15. Najm H, Hoomod HK, Hassan RJ, Sciences N. A Proposed Hybrid Cryptography Algorithm based on GOST and Salsa (20). Period Eng Nat Sci. 2020; 8(3): 1829-35.
16. Shaktawat R, Suwalka I, Lakshmi N, Panwar A. An Improved Encryption Technique for Digital Imaging and Communications in Medicine (DICOM). ECS Trans. 2022;107(1):8229. https://doi.org/10.1149/10701.8229ecst
17. Adnan M, Kalra S, Cresswell JC, Taylor GW, Tizhoosh HR. Federated learning and differential privacy for medical image analysis. Sci Rep. 2022;12(1):1953. https://doi.org/10.1038/s41598-022-05539-7 .
18. Jeong K, Lee Y, Sung J, Hong S. Security Analysis of HMAC/NMAC by using Fault Injection. J Appl Math 2013; 2013. https://doi.org/10.1155/2013/101907
19. Hao R, Li B, Ma B, Song L J. Algebraic Fault Attack on the SHA-256 Compression Function. J Appl Math. 2014; 4(2): 1-9. https://doi.org/10.7815/ijorcs.42.2014.079
20. Almazrooie M, Samsudin A, Singh . Improving the Diffusion of the Stream Cipher Salsa20 by Employing a Chaotic Logistic Map. J Inf Process Syst. 2015; 11(2): 310-24. https://doi.org/10.3745/JIPS.02.0024
21. Alshawi I, Muhalhal L. Improved Salsa20 Stream Cipher Diffusion Based on Random Chaotic Maps. Informatica . 2022; 46(7). https://doi.org/10.31449/inf.v46i7.4279
22. Zhang YJO. Cryptanalysis of a Novel Image Fusion Encryption Algorithm based on DNA Sequence Operation and Hyper-chaotic System. Optik . 2015; 126(2): 223-9. https://doi.org/10.1016/j.ijleo.2014.08.129

Published Online First: July, 2024
https://doi.org/10.21123/bsj.2024.9280
P-ISSN: 2078-8665 - E-ISSN: 2411-7986
Baghdad Science Journal

اقتراح طريقة للتشفير الخفيف للصور الملونة باستخدام SALSA20 مع دالة اشتقاق المفتاح

ابتهال طالب[1]، عبير سالم جميل[2] ، نداء فليح حسن [3]

[1]وزارة التعليم العالي والبحث العلمي ، بغداد ، العراق.
[2] قسم هندسة تقنيات الحاسوب ، كلية المنصور الجامعية ، العراق.
[3] قسم علوم الحاسوب ، الجامعة التكنولوجية ، بغداد ، العراق.

الخلاصة

نظرًا للتطور الاستثنائي في إنتاج وتبادل الوسائط المتعددة عبر الشبكات غير المحمية، اصبحت الحاجة لتطوير خوارزميات تشفير موثوقة وآمنة وفعالة أكبر. تقترح هذه الورقة خوارزمية جديدة للتشفير خفيفة الوزن تجمع بين SALSA20 ووظيفة اشتقاق المفتاح المستندة إلى كلمة المرور (2) PBKDF2) للصور الملونة. نظرًا لخصائص PBKDF2 مثل تعزيز أمان كلمة المرور من خلال تطبيق جولات متعددة من وظيفة عشوائية زائفة (PRF) لإنشاء المفتاح، يضيف استخدام Salt عشوائية إضافية إلى عملية اشتقاق المفتاح، لذلك يتم استخدامه لتعزيز SALSA20. ومن خلال توفير مساحة مفاتيح كبيرة، أثبت تشفير الصور خفيف الوزن الموصى به قوته ضد هجمات القوة الغاشمة. بالإضافة إلى ذلك، فإن الطريقة المقترحة قادرة على الحماية من الاختراق الإحصائي، والهجوم التفاضلي، ,والحفاظ على أمان الصورة استنادًا إلى الرسم البياني، والارتباط، وNPCR، وUACI، ومعايير الإنتروبيا. وقت التشفير لصورة الاختبار قليل جدًا، ويتراوح من ثانية إلى ثانية ونصف، لذا فهو قريب من التشغيل في الوقت الفعلي

الكلمات المفتاحية: التشفير الخفيف ,وظيفة الاشتقاق الرئيسية ,PBKDF2 ,وظيفة شبه عشوائية ,الهجمات , SALSA20, .SHA256