Cryptanalysis of Images Encrypted via Rubik's Cube Algorithm

Ömer Aras Kaplan, Ömer Buğrahan Çalışkan
Bilgisayar Mühendisliği Bölümü
Yıldız Teknik Üniversitesi, 34220 Istanbul, Türkiye
{l1117039, l1117076}@yildiz.edu.tr

***Özetçe* —Kolaylıkla ulaşabileceğimiz birçok görüntü şifreleme algoritmasının bulunması, aralarından güvenli bir algoritma bulmayı oldukça zorlaştırabilmektedir. Rubik Küpü Prensibine dayanan bir görüntü şifreleme algoritmasını makalemizin ana odağı olarak aldık. Bu makale, Kriptoloji ile ilgili giriş seviyesinde bilgiler ve algoritmanın analizi için kullanılan araçların tanıtımını sağlayacaktır. Ardından görüntü işleme algoritmasının çalışma aşamaları ortaya konacak ve bahsedilen araçlar kullanılarak algoritmanın ürettiği şifrelenmiş görüntüler test edilecektir. Makale sona ererken, bulgular tartışılarak analiz edilip yorumlanacaktır.**

Anahtar Kelimeler—Rubik Küpü Prensibi, Kriptoanaliz, Salt and Pepper, Kaba Kuvvet Saldırısı, Median Filtering, Cropping Attack Saldırısı

***Abstract*—Since there are many image encryption algorithms available, finding a secure one might prove quite challenging. This article focuses on a secure image encryption algorithm which is based on Rubik's Cube Principle. This paper provides introductory level information on cryptography and presents a brief overview of the tools used in the analysis of algorithm. Then, the paper puts the image encryption algorithm under the spotlights by explaining the flow of the algorithm and tests the encrypted images against different kinds of attacks. In the final sections, the discussion of the results are followed by the main conclusions drawn in this study. The aforementioned image encryption algorithm is called A Secure Image Encryption Algorithm Based on Rubik's Cube Principle[1].**

Keywords—Rubik's Cube Principle, Cryptanalysis, Salt and Pepper Noise, Brute Force Attack, Median Filtering, Cropping Attack

I. Introduction

The word Cryptography is coined from the Greek word "kryptos" (hidden) and "graphein" (to write). Cryptography is considered both as an art and a science field focusing on making communication intelligible only to the people who are intended to receive the information[2]. While it is hard to trace the first time cryptography was used in any manner, it is known that cryptology and encryption has been a part of humankind. The first clear method available is known as the Caesar cipher which was named after the Roman politician Julius Caesar, who had used the method in his private letters. The Caesar cipher is essentially a shifted alphabet which can be considered as a substitution cipher [3].

Since then, encryption methods have improved dramatically. Starting with the Renaissance Era, cryptography was taught and studied in depth. The methods grew more complicated and systhematic [4].

In 20th century, backed up by the waging world wars, demand for encryption and decryption techniques increased significantly. With the invention and development of computers and monitors, suddenly there was a new branch in cryptography called image encryption.

Today, there exists a number of different image encryption algorithms such as Elliptic Curve AES method, Arnold transform, DNA method, Wavelet Transform, Rubik Cube Transform, Fractional fourrier transform and Chaos systems [5].

Although many image encryption algorithms provide secure and reliable progress, their raw processing needs are generally higher than average[6]. One algorithm in particular, has piqued our interest with its resistance against brute force attacks and its resilient attitude against bit manipulation attacks. Furthermore, implementation of algorithm is relatively easy. Moreover, the algorithm is named after a real-life problem's solution while using a variation of that solution as means to encrypt the data. In light of these cases it was only logical for us to analyze this interesting subject[1].

From the information we gathered during our literature search, there are different ways to analyze the security and integrity of an image encryption algorithm. In the next part, the paper provides information about certain attacks that are useful for evaluating and analyzing an image encryption algorithm.

II. Tools Used in Testing

A. Salt and Pepper Noise

In an article about effects of removal attacks[7] it is stated that "salt and pepper" noise is basically a variation of statistical noise that uses a distinctly different probability density function. This noise shows gradual increase in pixel value variance and frequency component as the noise levels ascend, thus making the salt and pepper attacks a filter which makes it function as a high pass filter[7].

Given the fact that our subject only accepts gray scale images, using this attack may prove valuable for our findings.

B. *Cropping Image Attack*

Cropping Image attacks are generally used in watermark removal tests due to their ability to remove, edit or alter a specific area in an image. In this study, cropping image attack is used as a tool on encrypted images before decryption, in order to check what kind of effects it has on the decrypted image.

C. *Median Filter*

Median filtering is a commonly used image anti-forensic technique. Because of its nonlinear nature, median filtering can actually fix the traces of different attacks and altering attempts[8]. Median filter as a tool is used in this paper for cleaning noise and noise leftovers.

III. Description of Algorithm

A. *Pre-work Initializations*

I_0=a-bit gray scale image

M,N = Column and row size of image

$Iter_m$ = Maximum number of iterations

$Iter$ = Current value of iterations

B. *Encryption*

Step 1. Generate two arrays with size M and N respectively. These arrays will be called K_R and K_C. All elements of both arrays should get a random value ranging from 0 to 2^a.

Step 2. Determine $Iter_m$ and initialize $Iter$ as 0

Step 3. Increment $Iter$ by 1

Step 4. For each row(i) in I_0

Step 4.1. Compute the sum of all elements in the row i, this sum is denoted by α(i)

$$\alpha(i) = \sum_{j=1}^{N} I_0(i,j) \text{ i=1,2,3,4......M,}$$

Step 4.2 Compute modulo 2 of $\alpha(i)$, denoted by $M_{\alpha(i)}$

Step 4.3 Row i is left, or right, circular-shifted by K_R(i) positions (image pixels are moved by K_R(i) to the left or right direction, and the first pixel moves to the last pixel), according to the following:

if $M_{\alpha(i)}$= 0 then right circular shift

else left circular shift

Step 5. For each column j of image I_0,

Step 5.1. Compute the sum of all elements in the column j, this sum is denoted by $\beta(j)$,

$$\beta(i) = \sum_{i=1}^{M} I_0(i,j) \text{ j=1,2,3,4......N,}$$

Step 5.2. Compute modulo 2 of $\beta(j)$, denoted by M_β(j)

Step 5.3. Column j is down, or up, circular-shifted by $K_{C(i)}$ positions, according to the following:

if $M_{\beta(j)}$= 0 then, up circular shift

else down circular shift

Step 6 Using vector K_C, the bitwise XOR operator is applied to each row of scrambled image I_{scr} using the following expressions:

I_1(2i-1,j) = I_{scr}(2i-1,j) $\bigoplus$ K_R(j)

I_1(2i,j)=I_{scr}(2i,j) $\bigoplus$ rot180(K_R(j)).

where $\bigoplus$ and rot 180(K_C) represent the bitwise XOR operator and the flipping of vector K_C from left to right, respectively.

Step 7. Using vector K_R, the bitwise XOR operator is applied to each column of image I_1 using the following formulas:

I_{ENC}(i,2j-1) = I_1(i,2j-1) $\bigoplus$ K_R(j)

I_{ENC}(i,2j)=I_1(i,2j) $\bigoplus$ rot180(K_R(j)).

with rot 180(K_R) indicating the left to right flip of vector K_R.

Step 8. if Iter= $Iter_m$, then encrypted image I_{ENC} is created and encryption process is done; otherwise, the algorithm branches to step 3.

Note that K_R and K_C are considered keys for decryption

C. *Decryption*

Step 1. Initialize Iter=0

Step 2. Increment the counter by one Iter= Iter+1

Step 3. The bitwise XOR operation is applied to K_R and each column of the encrypted image I_{ENC} as follows

I_1(i,2j-1)=I_{ENC}(i,2j-1) $\bigoplus$ K_R(j)

I_1(i,2j)=I_{ENC}(i,2j)=I_{ENC}(i,2j) $\bigoplus$ rot180(K_R(j))

Step 4. Then using the K$_C$, the bitwise XOR operator is applied to each row of image I$_1$

I$_{SCR}$(2i-1,j)=I$_1$(2i-1,j) $\bigoplus$ K$_C$(j)

I$_{SCR}$(2i,j)=I$_1$(2i,j) $\bigoplus$ rot180(K$_C$(j))

Step 5. For each column j on the scrambled image I$_{SCR}$

Step 5.1. Compute the sum of all elements in that column j, denoted as β_{SCR}(j):

$$\beta_{SCR}(i) = \sum_{i=1}^{M} I_{SCR}(i,j) \text{ j=1,2,3,4......N,}$$

Step 5.2. Compute modulo 2 of β_{SCR}(j), denoted by M$_{\beta SCR(}j)$

Step 5.3 column j is then down or up, circular shifted by K$_C$(i) according to the following:

if M$_{\beta SCR(j)}$= 0 up circular shift

else down circular shift

Step 6. For each row i of scrambled image I$_{SCR}$,

Step 6.1. Compute the sum of all elements in that row i, denoted as α_{SCR}(i):

$$\alpha_{SCR}(i) = \sum_{j=1}^{N} I_{SCR}(i,j) \text{ i=1,2,3,4......M,}$$

Step 6.2. Compute modulo 2 of α_{SCR}(j), denoted by M$_{\alpha SCR(}j)$

Step 6.3 row i is then left or right, circular shifted by K$_R$(i) according to the following:

if M$_{\alpha SCR(j)}$= 0 right circular shift

else left circular shift

Step 7. If Iter= Iter$_m$, then image I$_{ENC}$ is decrypted and the decryption process is done; otherwise the algorithm branches back to step 2 [1].

IV. Results & Findings

A. Brute-Force Attacks

For ease of understanding, let M and N be the column and row size of the image. The algorithm suggests that there should be two arrays with sizes of M and N. Then every element inside them should get a value from 0 to 2^{α}. That means there would be $2^{\alpha.M}$ possible value for each column. Combining that with the count of rows, the formula for the size of our space set should be $2^{\alpha.(M+N)}$.

Grayscale images usually have the α rating of 8. Even a small picture with a size of 16 x 16 would make the space set as big as 256^{16+16}=2^{256} different set of values for K$_R$ and K$_C$. For a common computer with reasonable specs it takes about 0.15 seconds to decrypt an image. Given that the space is 2^{256} , it would take

$$2^{256} = 1.1579209e + 77$$

different brute force combinations to try all possibilities.

And it would take

$$1.1579209e + 77.0, 15 = 1, 7368814e + 76$$

Seconds to complete the task.

After some calculation the amount of time needed to complete this task in years is:

$$3.669308.10^{69}$$

Years should pass before the computer finishes its task.

The length of this test is dependent on the host computer's power and the regression used in the brute force algorithm. Even though it is an image of 16x16 pixels it would take a really long time to crack it using brute force attacks. Thus it can be said that this algorithm is fairly robust against brute force attacks.

B. Salt & Pepper Attacks

In this section, we used salt and pepper attacks on encrypted images. Then we decrypted those images to see the differences between the original and attacked images. The paper provides the correlation values of the original and attacked images as well as the correlation values after median filter has been applied to remove the noises.

Figure 1 Original Decrypted Lena

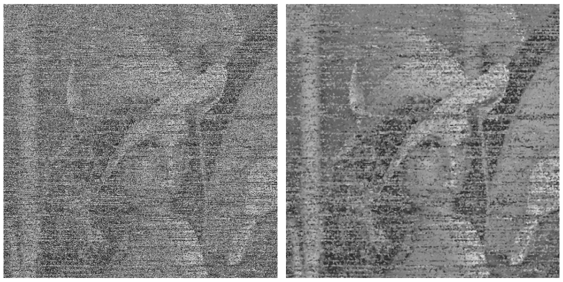

Figure 2 Decrypted Salt and Pepper Lena and Median Filtered Decrypted Salt and Pepper Lena

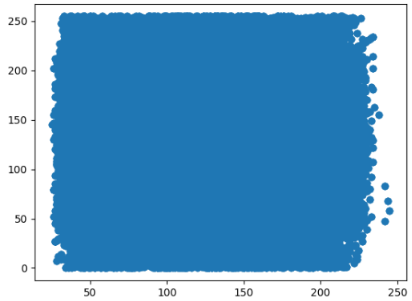

Figure 3 Correlation Between Decrypted Lena and Decrypted Salt and Pepper Lena

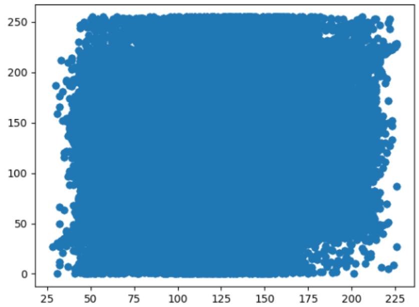

Figure 4 Correlation Between Decrypted Lena and Median Filtered Decrypted Salt and Pepper Lena

Salt and pepper method used in this section has a noise frequency value of 0,9. As the number goes down, there will be more noise in the image. Which will incur less resemblance to the original image.

Correlation value between decrypted lena and Salt and Pepper lena without median filtering is 0,2121. Generally salt and pepper attacks have less frequency value than 0,9. In our studies lesser frequencies corrupted the image exponentially as the value gone down.

However using median filtering nearly doubled correlation value from 0,2121 to 0,3852. Which means the algorithm actually responds to median filtering quite good.

C. Cropping Image Attacks

In this section, we used cropping image attack on encrypted images. We cropped %15 of the image in the middle after encryption process. The paper provides correlation values of original and attacked images as well as the correlation values after median filter has been applied to remove the noises.

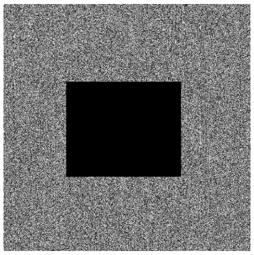

Figure 5 Cropping Attack on Encrypted Image

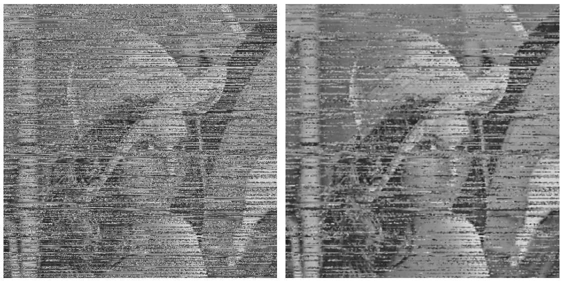

Figure 6 Decrypted Cropped Lena and Median Filtered Decrypted Cropped Lena

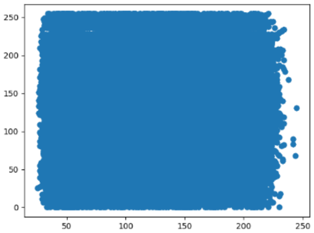

Figure 7 Correlation Between Decrypted Lena and Decrypted Cropped Lena

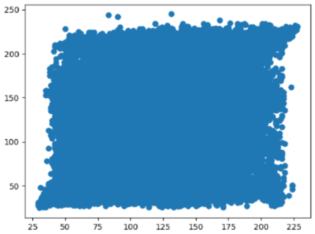

Figure 8 Correlation Between Decrypted Lena and Median Filtered Decrypted Cropped Lena

The encrypted image was cropped in the middle by %15. After the decryption correlation value between the original image and decrypted cropped image is 0,3421. Comparing this value to the one we had in salt and pepper attacks, we can say that the algorithm actually shows good resilience against cropping attacks.

Moreover, when the decrypted cropped image is median filtered the correlation value jumps up to 0,6387. This case complements what had been provided in the previous section about median filtering on this algorithm.

V. Conclusion

In this paper we analyzed a proposed image encryption algorithm for its integrity and security. The algorithm is called A Secure Image Encryption Algorithm Based on Rubik's Cube Principle. The algorithm permutates image pixels then applies XOR operations on rows and columns using randomly generated key arrays. We conducted Salt and Pepper Attack, Cropping Attack and Median Filtering to test the integrity of images encrypted with this algorithm in addition to the algorithm's theoretical analysis against Brute-Force Attacks. The algorithm shows good robustness against Cropped Image Attacks while it is only secure to a certain degree against Salt and Pepper Attacks. Median Filtering generally provides adequate results when cleaning or reverting manipulated pixels.

In conclusion, the algorithm, while being fast and usable, still retains security and integrity in an accepted level.

References

[1] A. B. Khaled Loukhaoukha, Jean-Yves Chouinard, "A secure image encryption algorithm based on rubik's cube principle," *Journal of Electrical and Computer Engineering*, vol. 2012, no. 12, p. 13, 2012.

[2] L. D and P. G, "Cryptology: From caeser ciphers to public-key cryptosystem," *The College Mathematics Journal*, vol. 18, no. 1, p. 2, 1987.

[3] S.-G. O. Abraham, O., "An improved caesar cipher (icc)algorithm," *International Journal Of Engineering ScienceAdvancedTechnology (IJESAT)*, vol. 2, no. 12, pp. 1198–1202, 2012.

[4] D. Davies, "A brief history of cryptography," *Information Security Technical Report*, vol. 2, no. 2, pp. 14–17, 1997.

[5] P.-P. I. A. M. Geetha, S., "A literature review on image encryption techniques," *International Journal of Information Security and Privacy (IJISP)*, vol. 2012, no. 12, pp. 42–83, 2018.

[6] F. Sun, S. Liu, Z. Li, and Z. Lü, "A novel image encryption scheme based on spatial chaos map, chaos, solitons fractals,," *Journal of Electrical and Computer Engineering*, vol. 38, no. 3, pp. 631–640, 2008.

[7] S. . M. M. Song, Chunlin Sudirman, "A spatial and frequency domain analysis of the effect of removal attacks on digital image watermarks," *Journal of Electrical and Computer Engineering*, vol. 2010.

[8] S. A. V. J. M. M. A. . E. S. Sharma, S., "Anti-forensic technique for median filtering using l1-l2 tv model," *Journal of Electrical and Computer Engineering*, vol. 2016, 2012.