Available online @ https://jjem.jnnce.ac.in
https:www.doi.org/10.37314/JJEM. 2022.060108
Indexed in International Scientific Indexing (ISI)
Impact factor: 1.395 for 2021-22
Published on: 31January 2022

An approach to achieve message integrity using SHA-256 and Elliptic Curve Cryptography with Salting

Ganavi M[1]*, Prabhudeva S[2]

[1*]Department of Computer Science & Engineering, [2]Dept. of MCA
JNN College of Engineering, Shivamogga

gaanavi4@jnnce.ac.in, pdshirematt@gmail.com

Abstract

In this digital world, securing the integrity of the information plays an important role. For a modernized venture, information integrity is requisite for the veracity & competence of employment processes and decision-making. Integrity verification is the one where the information has not been tampered with while it is stored or communicated. Hash function such as SHA-256 is the strongest function, can be used to achieve information integrity. In this paper, a method is proposed where the images are hashed using SHA-256, scrambled using Elliptic Curve Cryptography (ECC), and then added the salted values to generate the secret image. ECC helps achieve high security with a smaller key size. Salting is applied to the scrambled image to increase the randomness. This makes it difficult for the attacker to identify the secret image and reverse back the hashing as hashing is a one-way function. Simulation outcomes of the proposed method have been analyzed by measuring Peak Signal to Noise Ratio (PSNR), Mean Squared Error (MSE), Structural Similarity Index (SSIM), Number of Pixel Change Rate (NPCR), and Unified Average Changing Intensity (UACI). The proposed method is strong against the differential attacks.

Keywords: *Integrity, Elliptic Curve Cryptography, SHA 256, Scramble, Descramble, Salting*

1. Introduction

Information integrity indicates its veracity and firmness saved in the directory, information repository, and information exchange. Image information security [5] is very much predominant, mainly in the military, commercial and medical fields. Such applications include patient data at the hospital, reports of diagnostic, patient's health, and laboratory tests. The information integrity is always used to verify its quality. It is the guarantee used to check that digital information is uncorrupted. It can only be altered by the authorized user who has permission to do so. Integrity prevents the information from illegal diversification.

Cryptography can be applied to obtain the goals of information security, including confidentiality, integrity, and authentication [2, 11]. It can also be used to guarantee the integrity of information by applying the use of hashing functions and message digests. Hashing functions calculate the fixed-sized output for the variable size input information. Generated output size is reduced. A valuable hashing function should show a property known as the avalanche effect, where the generated hash output should change fully for a small change at the input. Generated hash is in hexadecimal form and it is a one-way process so that it can not be reversed back to get back the original information. It this become difficult for the hackers to extract back the actual information.

A better hash function should never generate the same hash value for two different input

information [6]. If so, then it is called a hash collision. A hashing can only be given priority as best and tolerable if it can bid a very low opportunity of collision. The most acceptable type of hashing used for file integrity checks is MD5, SHA-2, and CRC32.

Elliptic Curve Cryptography (ECC) is a present-time public-key crypto-system well known for small key-size, speedy, and more structured than the other asymmetric cryptosystems [9]. ECC is used to scramble the information so that only authorized parties can descramble it by applying its private key. ECC has been a recent analysis topic within the space of information security. ECC is the next generation of public key cryptosystem and, imparts a remarkably more secure system than first-generation public key cryptosystem than RSA. If anxious about guaranteeing the highest level of security while conserving performance, ECC is the best choice to adopt. ECC is suitable for key mediation, digital signatures, and pseudo-random generators. Authentication approaches based on ECC provides better security for information communication through mobile phones, smart cards, financial transaction, and sensitive passwords.

An encryption method of digital color-image using elliptic curve cryptography El Gamal encryption has been proposed in [4]. Also used a 3D Lorenz map. A block-based ECC is used first, followed by a chaotic sequence generator has been proposed in [1]. Successful to achieve many security measures. An encryption method [7] where the input image is divided into five planes and each is encrypted using ECC. The hyperchaotic sequence is used in the permutation stage.

A hash of salt merged with passwords has been proposed [10]. Hashes are at risk of birthday attacks. This system is used to generate multiple passwords for a single combination of salt and password. Salt with passwords is dealt with Davies-Meyer, a compression function. Also dealt with an ofñine birthday forgery attack. With all these existing approaches, the motivation of this paper is to use the salting technique along with hashing and scrambling to further increase the integrity of the images. A scrambling algorithm is used to randomize the images which is helpful to achieve confidentiality. An approach is proposed where it combines hashing and scrambling methods. The main objective is to achieve integrity for the input image. Hashing, SHA-256 [3, 8, 12] is so strong that, even though the hacker gets a hash value, it is not possible to revert the original information. The technique of salting plays an important role in the proposed method. Salting makes hash value different for the same hash value with the same input information. This makes it difficult for the third party to extract back the actual information.

1. Proposed method

The proposed approach contains three stages, shown in Figure 1. In the first stage, a hash of an image is generated by using SHA-256. In the second stage, the hash value is scrambled using ECC to generate the cipher. Scrambling is carried out by ECC where the public key is used for scrambling and the private key is used at the descrambling process. At the last stage, the scrambled hash is randomized using the salting technique. Salting generates random numbers, and these random numbers are added to the scrambled image. As salting generates random numbers which are completely different each time. Hashing generates the same hash value for the same input. But in the proposed method, an attempt is made to generate a completely different hash value after scrambling and salting technique. So, it is difficult for a hacker to identify the hashed value and try to extract the input image. A reverse procedure is applied at the receiver side. The descrambled image is checked for integrity. If the hash value of the descrambled image is the same as the hash value found at the initial stage, then it indicates the image is not modified.

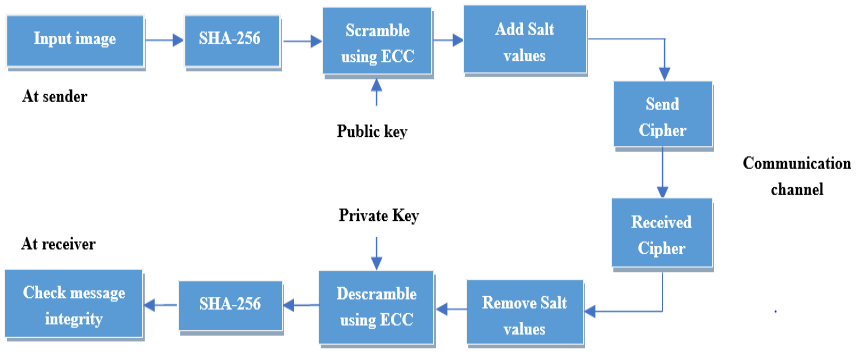

Figure 1: Block diagram of the proposed system

Table 1: Index values for the elliptic curve Ep (a, b)

Pixel value	Points on ECC
0	(0,32)
1	(0,219)
2	(1,5)
. .	
5	(2,6)
.	.
25	(22,28)
.	.
62	(59,153)
.	.
78	(74,210)
.	.
105	(101,118)
.	.
201	(191,49)
.	.
255	(239,1)

At the sender, the steps followed are as below

i. Read input color image '*I'* & separate R, G, B channels (*Ir, Ig, Ib*) from the input image.

ii. Read '*Ir'* of size (M_1 *X* M_2) & compute the hash of an R-channel image '*Ir'* by using SHA-256.

$$Ir\_hash = sha256hasher.ComputeHash(Ir) \quad (1)$$

iii. Scramble 32-byte hash using ECC by considering an elliptic curve Ep (a, b) and find all possible points as shown in Table 1.

iv. Choose generator *G (x1, x2)* with large order *n* in *Ep (a, b)*

v. At the sender**,** choose a private key and compute the public key using generator *'G'* as

$$Pr1 = x \quad (2)$$

$$Pb1 = X = x * G(x1, x2) = (Pb1x, Pb1y) \quad (3)$$

vi. Choose another random number 'rd' and multiply with 'G'

$$Rd = rd * G(x1, x2) = (Rdx, Rdy) \quad (4)$$

vii. Read each byte of hash value *'Ir_hash'* and map to the corresponding point which are generated in Table 1. Repeat this process for all the bytes of the hash value.

$$Ir(x1, x2) = ECC(Ir\_hash) \quad (5)$$

viii. Compute cipher by considering each value in *'Ir(x1,x2)'* using *'Pr1'* and '*Pb2'* as

$$Ici(x1, x2) = Ir(x1, x2) + (Pr1 * Pb2) + (Rd) \quad (6)$$

where $i = 0, 1, 2, \ldots\ldots M1$

ix. The newly generated '*Ici(x1,x2)'* is again mapped to a new value according to the index table 1, resulting in a final scrambled image '*Ic'*.

$$Ic = ECC(Ici(x1, x2)) \quad (7)$$

x. Generate salt values *'Isalt'* and add it to scrambled image *'Ic'* obtaining *'Icr'*

$$Icr = Isalt + Ic \quad (8)$$

xi. Repeat this process for G- and B-channel images *'Ig' & 'Ib'*

xii. Combine ciphers of all three channels and send *'Icipher'* & salt values *'Isalt'* to the receiver.

$$Icipher = Icr + Icg + Icb \quad (9)$$

At the receiver, the steps followed are as below

i. Receive Cipher image *'Icipher' & 'Isalt'*

ii. Separate R, G, B channels (CI*r, CIg, CIb*) from the cipher image.

iii. Read R-channel image *'CIr'* and subtract salt values from *'CIr'*

$$CCc = CIr - Isalt \quad (10)$$

iv. Choose a private key and compute the public key using generator 'G' as

$$\Pr 2 = y \quad (11)$$

$$\mathrm{Pb2} = Y = y * G\,(x1, x2) = (Pb2x, Pb2y) \quad (12)$$

v. Descramble *'CCc'* using ECC to get back the actual plaintext using

$$Pcr\,(x1, x2) = CCc(x1, x2) - (\Pr 2 * \mathrm{Pb1}) - (Rd) \quad (13)$$

vi. Repeat this process for G- and B-channel images *'Pcg' & 'Pcb'*

vii. Combine descrambled of all three channels.

$$Pc = Pcr + Pcg + Pcb \quad (14)$$

viii. Compute the hash of an R-channel image '*Pcr'* by using SHA-256.

$$Pcr\_hash = sha256hasher.ComputeHash(Pcr) \quad (15)$$

ix. Perform integrity check by comparing the hash of the '*Pcr_hash' and 'Ir_hash'*. If the hash remains the same, then it indicates that the input image is not modified by the third party and it can be accepted. Otherwise, it is rejected.

x. Repeat this process for G- and B- channel hash values.

5. Results and Performance Analysis

The set of input secret images used for the proposed work is shown in Figure 2. Input secret images are used from USC-SIPI [13] and KODAK [14] Image datasets. Figure 3 shows the ECC plot based on the points generated for a=4, b=20, p=251. Hash of the input image, ECC scrambled hash, salt generated, salted hash, and hash from descrambled are shown in Figure 4. (a), (b), (c), (d) & (e). Histograms of all these are presented in Figure 5. It is observed that the histogram of the scrambled with salt indicates the uniform distribution of information, which makes any viewer judge the tonal distribution. This shows more randomness is achieved from the proposed approach. The Hash of the descrambled with salt and stored hash is the same, it indicates that the transmitted image has not been modified by any hacker. Even if a single bit in the communicated data is changed, then the calculated hash will be completely different. So, the proposed system is successful to achieve message integrity checks.

Figure 2: Set of input color images

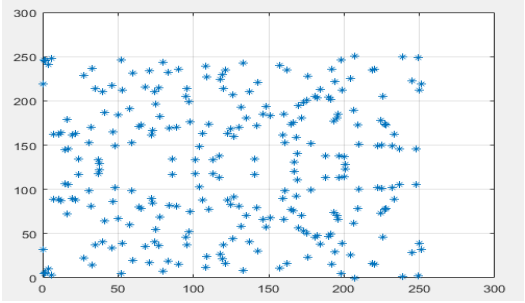

Figure 3: Plot of ECC points for a=4, b=20, p=251

53, 22, 7A, B8, 32, D1, 09, E0, 6F, 03, 59, 93, DB, AE, '1B, FA, 82, FC, BE, 6A, 2A, 9A, A5, 10, 10, C4, 4E, 44, C5, 9A, 85, CC (a)
6, 199, 0, 152, 74, 141, 149, 202, 37, 125, 93, 253, 140, 107, 48, 5, 157, 146, 115, 175, 45, 10, 130, 167, 167, 169, 27, 190, 118,10, 87, 119 (b)
131, 127, 127, 129, 126, 130, 128, 128, 128, 128, 129, 131, 127, 126, 127, 130, 129, 125, 127, 131, 126, 128, 128, 127, 129, 126, 129, 129, 128, 127, 125, 128 (c)
137, 255, 127, 255, 200, 255, 255, 255, 165, 253, 222, 255, 255, 233, 175, 135, 255, 255, 242, 255, 171, 138, 255, 255, 255, 255, 156, 255, 246, 137, 212, 247 (d)
53, 22, 7A, B8, 32, D1, 09, E0, 6F, 03, 59, 93, DB, AE, '1B, FA, 82, FC, BE, 6A, 2A, 9A, A5, 10, 10, C4, 4E, 44, C5, 9A, 85, CC (e)

Figure 4: (a) Hash value, (b) Scrambled
(c) Salt value, (d) Salted hash (e) Descrambled hash

Figure 5: Original input image, Histograms of input, ciphers of scrambled (R-, G- & B-channels) & descrambled images.

5.1. Security analysis

1. Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR), and Structural Similarity Index (SSIM)

MSE is used to evaluate the error among estimated and actual values. Higher MSE gives better image security for the scrambled image.

$$MSE = \frac{1}{P1*P2}\sum_{i=1}^{P2}\sum_{j=1}^{P1}\left[J1(i,j) - J1'(i,j)\right]^2 \quad (16)$$

where,
P_1, P_2 = Total rows & columns in picture,
$J1(i, j)$ = input picture and
$J1'(i, j)$ = output picture

PSNR is used for the computation of quality assessment among images. Lower PSNR represents more randomness in the scrambled image.

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right) \quad (17)$$

An SSIM measure assesses the perceptual dissimilarity among two indistinguishable images. In a scrambled image the SSIM value should be nearer to zero.

$$I(x,y) = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu^2{}_x + \mu^2{}_y + c_1\right)\left(\sigma^2{}_x + \sigma^2{}_y + c_2\right)} \quad (18)$$

where c_1 and c_2 are constants, x and y are cover & cipher images. Then, μ & σ are average & standard deviation.

2. Number of Pixel Change Rate (NPCR) and Unified Average Changing Intensity (UACI)

NPCR is the difference in the pace of the number of pixels in cipher when just a single pixel of an input image is adjusted. NPCR should be nearer to 100 for scrambled.

$$NPCR = \frac{1}{P*Q}\sum_{w1=1}^{P}\sum_{w2=1}^{Q} A(w1,w2)*100\% \quad (19)$$

UACI determines the average intensity variation among two images [27].

$$UACI = \left(\sum_{w1=1}^{P}\sum_{w2=1}^{Q}\left(\frac{|y1(w1,w2) - y1'(w1,w2)|}{255*P*Q}\right)\right)*100\% \quad (20)$$

where P*Q is image dimension, *y1(w1, w2)* is

ciphertext pixel concerning original plaintext, and *y1'(w1, w2)* is ciphertext pixel concerning changed plaintext.

Lesser PSNR & more MSE gives more scrambling of hash values indicating more randomness in the scrambled hash. This becomes difficult for any intruder to reverse back the process. For scrambled hash, always SSIM should be very less, NPCR should be nearer 100, and UACI ideal value is more than 33.46. These values are presented in Tables 2 & 3. Five different images are taken from the USC-SIPI & Kodak database. The last row indicates the average value for all the input images. For the proposed method, PSNR as 4.87664 (dB), MSE as 2.55E+04, SSIM as 0.0562, NPCR as 99.8778, and UACI as 33.4635 respectively. The values for performance metrics are satisfying w.r.t to their ideal values.

Table 2: PSNR, MSE and SSIM for scrambled hash

Input images	PSNR (dB)	MSE	SSIM
Peppers.tiff	3.8385	2.6867e+04	0.0392
San Diego.tiff	11.033	5.1259e+03	0.0711
4.1.01.tiff	2.7844	3.4247e+04	0.0487
kodim02.png	2.4384	3.7088e+04	0.0315
kodim03.png	4.2889	2.4220e+04	0.0905
Average	4.8766	2.55e+04	0.0562

Table 3: NPCR & UACI for scrambled images

Input images	NPCR	UACI
Peppers.tiff	99.9023	33.4635
San Diego.tiff	99.7299	33.4635
4.1.01.tiff	99.9282	33.4635
kodim02.png	99.9526	33.4635
kodim03.png	99.8764	33.4635
Average	99.8778	33.4635

Comparison of PSNR & MSE of the proposed method with the method in [4] is presented in Table 4 and the same is plotted in Figure 6. It is observed that PSNR is lesser, and MSE is more when compared to the method in [4]. Therefore, the proposed method provides more randomness for the scrambled hash, making it more complicated for the hacker.

Table 4: Comparison of PSNR, MSE & SSIM for scrambled image (peppers.tiff)

Methods	PSNR	MSE
Dhanesh Kumar, et al [4]	8.1516	1.0145e+04
Proposed method	3.8385	2.6867e+04

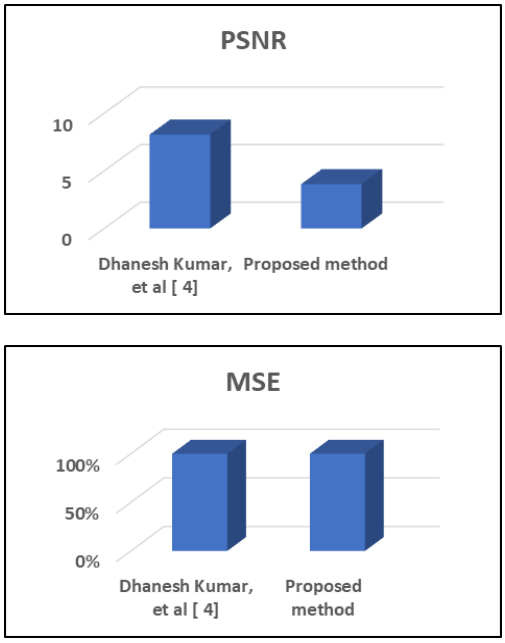

Figure 6: A plot of PSNR & MSE for the proposed method when compared to the method [4].

Comparison of NPCR & UACI for the proposed method (Peppers.tiff as input image) with other methods in [1] & [7] are tabulated in Table 5 and the same is plotted in Figure 7. It is observed that NPCR is higher than the methods in [1] & [7] and UACI is >33%. This represents even if a single bit at the input is changed, the entire scrambled hash is changed. Therefore, the proposed method is strong against differential attacks.

Table 5: Comparison of NPCR, UACI for scrambled image (peppers.tiff)

Methods	NPCR (%)	UACI (%)
Liang et al [7]	99.6124	33.4600
Abdelfattah [1]	99.60	33.44
Proposed method	99.8778	33.4635

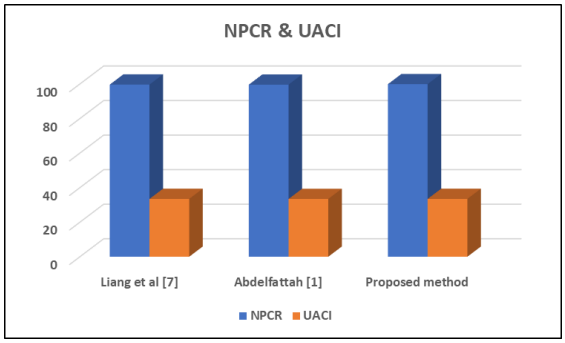

Figure 7: A plot of NPCR & UACI for the proposed method when compared to the methods [7,1].

6. Conclusion and Future Scope

In this proposed method, SHA-256, the ECC, and the Salting technique are combined to strengthen the integrity of the images. SHA-256 is applied on input to produce the hash of an input image. Mapping operations on the hashed value for the elliptic curve points are carried out. Combining hashing with the ECC resulted in a scrambled hash. Salt values are added to scrambled hash outputs to show that the algorithm gives better statistical measures. Now, the scrambled hash is completely different for the same input with the same hash. The performance analysis for scrambled is carried out by measuring PSNR, MSE, SSIM, NPCR, and UACI. The proposed method gives good results for the performance metrics PSNR, MSE, NPCR & UACI when compared to other methods [1], [4] & [7]. Histogram of scrambled shows that salting is helpful to achieve more randomness. This approach is progressively secure and suitable for applications of hashing &scrambling images. In the future, the work can be extended to apply on the most significant bit planes of the image as the least significant bit planes contain noise. This reduces the total execution time.

References

1. Abdelfatah, Roayat Ismail, Secure image transmission using chaotic-enhanced elliptic curve cryptography, IEEE Access, Vol. 8, 2019, pp. 3875-3890.

2. Benssalah, Mustapha, Yesser Rhaskali, and Karim Drouiche, An efficient image encryption scheme for TMIS based on elliptic curve integrated encryption and linear cryptography, Multimedia Tools and Applications, Vol. 80, No. 2, 2021, pp. 2081-2107.

3. Bhadke, Abhilash Ashok, Surender Kannaiyan, and Vipin Kamble, Symmetric chaos-based image encryption technique on image bit-planes using SHA-256, Proceedings of the 24th National Conference on Communications (NCC), 25-28 February 2018, Hyderabad, India, pp. 1-6. Publisher: IEEE, DOI: 10.1109/NCC.2018.8600222

4. Dhanesh Kumar, Anand B. Joshi, Sonali Singh, Vishnu Narayan Mishra, Digital color-image encryption scheme based on elliptic curve cryptography ElGamal encryption and 3D Lorenz map, In AIP Conference Proceedings, Vol. 2364, Issue 1, pp. 020026-1 to 020026-13, 2021. DOI:10.1063/5.0062877

5. Gul, Ertugrul, and Serkan Ozturk, A novel hash function based fragile watermarking method for image integrity, Multimedia Tools and Applications, Vol. 78, No. 13, 2019, pp. 17701-17718.
https://doi.org/10.1007/s11042-018-7084-0

6. Hambouz, Ahmed, Yousef Shaheen, Abdelrahman Manna, Mustafa Al-Fayoumi, and Sara Tedmori, Achieving data integrity and confidentiality using image steganography and hashing techniques, Proceedings of the 2nd International Conference on new Trends in

Computing Sciences (ICTCS), Amman, Jordan, 09-11 October 2019, pp. 1-6. Publisher: IEEE.
DOI: 10.1109/ICTCS.2019.8923060

7. Liang, Haotian, Guidong Zhang, Wenjin Hou, Pinyi Huang, Bo Liu, and Shouliang Li, A Novel Asymmetric Hyperchaotic Image Encryption Scheme Based on Elliptic Curve Cryptography, Applied Sciences, Vol. 11, No. 12, 2021. https://doi.org/10.3390/ app11125691

8. Liao, Xiaofeng, Ayesha Kulsoom, and Sami Ullah, A modified (Dual) fusion technique for image encryption using SHA-256 hash and multiple chaotic maps, Multimedia Tools and Applications, Vol. 75, No. 18, 2016, pp. 11241-11266.

9. Luo, Yuling, Xue Ouyang, Junxiu Liu, and Lvchen Cao, An image encryption method based on elliptic curve elgamal encryption and chaotic systems, IEEE Access, Vol. 7, 2019, pp. 38507-38522.

10. P Gauravaram, Security Analysis of salt|| password Hashes, In 2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT), 26-28 November 2012, Kuala Lumpur, Malaysia, pp. 25-30, Publisher: IEEE.
DOI:10.1109/ ACSAT.2012.49

11. Patil, Pooja, and Rajesh Bansode, Performance Evaluation of Hybrid Cryptography Algorithm for Secure Sharing of Text & Images, International Research Journal of Engineering and Technology (IRJET), Vol. 7, No. 9, 2020, pp. 3773-3778.

12. Zhu, Shuqin, Congxu Zhu, and Wenhong Wang, A new image encryption algorithm based on chaos and secures hash SHA-256, Entropy, Vol. 20, No. 9, 2018. https://doi.org/10.3390/e20090716

13. USC-SIPI Image Database Website. http://sipi.usc.edu/database

14. KODAK Image Dataset Website. http://r0k.us/graphics/kodak