Arabian Journal for Science and Engineering (2021) 46:3581–3593
https://doi.org/10.1007/s13369-020-05196-w

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

Secure CT-Image Encryption for COVID-19 Infections Using HBBS-Based Multiple Key-Streams

Omar Reyad[1,2] · **Mohamed Esmail Karar**[1,3]

Received: 16 August 2020 / Accepted: 30 November 2020 / Published online: 5 January 2021

Abstract
The task of preserving patient data is becoming more sophisticated with the evolution of technology and its integration with the medical sector in the form of telemedicine and electronic health (e-health). Secured medical image transmission requires adequate techniques for protecting patient privacy. This study aims at encrypting Coronavirus (COVID-19) images of Computed Tomography (CT) chest scan into cipherimages for secure real-world data transmission of infected patients. Provably safe pseudo-random generators are used for the production of a "key-stream" to achieve high privacy of patient data. The Blum Blum Shub (BBS) generator is a powerful generator of pseudo-random bit-strings. In this article, a hashing version of BBS, namely Hash-BBS (HBBS) generator, is presented to exploit the benefits of a hash function to reinforce the integrity of extracted binary sequences for creating multiple key-streams. The NIST-test-suite has been used to analyze and verify the statistical properties of resulted key bit-strings of all tested operations. The obtained bit-strings showed good randomness properties; consequently, uniform distributed binary sequence was achieved over the key length. Based on the obtained key-streams, an encryption scheme of four COVID-19 CT-images is proposed and designed to attain a high grade of confidentiality and integrity in transmission of medical data. In addition, a comprehensive performance analysis was done using different evaluation metrics. The evaluation results of this study demonstrated that the proposed key-stream generator outperforms the other security methods of previous studies. Therefore, it can be successfully applied to satisfy security requirements of transmitting CT-images for COVID-19 patients.

Keywords Blum blum shub · Hash function · Key-stream generator · Image encryption · Coronavirus COVID-19 and security analysis

1 Introduction

Coronaviruses belong to perilous viruses, which can cause serious infectious diseases [1]. The pandemic of the novel Coronavirus 2019 (COVID-19) was begun in December 2019 in Wuhan city, China, and rapidly infected people in other countries around the world [2, 3]. On 22 October 2020, the weekly epidemiological and operational updates on COVID-19 of the World Health Organization (WHO) stated that the total number of confirmed infectious cases is globally 40,890,712 and the deaths become 1,126,351 [4].

According to the WHO scientific guidelines for national laboratories, the Real-Time Polymerase Chain Reaction (RT-PCR) assay is the gold standard for verifying the diagnosis of COVID-19 [5]. However, the RT-PCR test becomes insufficient, and showed high levels results of false-negative to confirm positive COVID-19 cases. Therefore, medical imaging techniques, for example, chest X-ray and Computed Tomography (CT), have been considered as a powerful tool to detect COVID-19 infections, especially in emerging cases of pregnant women and children [6]. Furthermore, volumetric CT-thorax images for soft tissues and lungs have been verified in the current studies to assist clinicians to identify COVID-19 patients [7]. The main features of CT-images for patients with COVID-19 include Ground-Glass Opacity

✉ Mohamed Esmail Karar
mekarar@ieee.org; mkarar@su.edu.sa

Omar Reyad
oreyad@su.edu.sa

1 College of Computing and Information Technology, Shaqra University, Shaqra, Saudi Arabia

2 Faculty of Science, Sohag University, Sohag 82524, Egypt

3 Faculty of Electronic Engineering, Menoufia University, Menoufia, Egypt

(GGO), multifocal patchy consolidation, and distributed lesions in the peripheral part of lungs [8].

In parallel with recent medical imaging modalities and rising cybertechnology of communications, the healthcare involved these technological advancements by employing the Internet to transfer the data of patients [9]. Therefore, it is highly important to protect the privacy of patients by encrypting their transmitted data over the general computer networks. Techniques of watermarking, steganography and cryptography present a vital role to accomplish the required tasks of securing patient data, including different technologies of medical imaging such as CT and Magnetic Resonance Imaging (MRI) [10].

The target images in this study are medical CT-images motivated by the current COVID-19 pandemic. The presented work focuses on applying advanced encryption techniques to safeguard CT-images of confirmed COVID-19 patients from unauthorized access. In addition, the encryption algorithm can be applied to any type of images to achieve high-level of privacy and security requirements.

1.1 Contribution of this Study

Due to the importance of patient's privacy protection, this paper presents a new Hash-based BBS (HBBS) pseudo-random bit generator to achieve integrity and security of COVID-19 patient data. The proposed scheme leads to a protected medical image data transmission system that thwart against privacy breaches. For integrity purposes, the hash value is generated using Secure Hash Algorithm SHA-256 and is hidden in the Least Significant Bit (LSB) of the extracted pseudo-random bits in the purpose of generating multiple key-streams. Also, an encryption technique is proposed and designed to accomplish a high grade of security in transmission of medical data. The simulation analysis demonstrated that the HBBS generator has large key space and can satisfy the performance requirements for the confidentiality of CT-images of COVID-19 patients.

For image encryption methods, a range of consistency tests is assessed. They are all bivariate, exploiting the differences between corresponding pixels in the original and encrypted images. It is shown that while certain numerical measures correlate well with the response of the observers to a given encryption technique, they are not accurate for an assessment of various techniques. Here, multiple metrics and analysis have been applied to the resulted cipherimages to ensure the effectiveness of our proposed scheme. Entropy, Peak Signal-to-Noise Ratio, Structural Similarity, Mean Absolute Error, Mean Square Error, Unified Average Change Intensity, Number of Pixel Change Rate and Correlation Coefficient Analysis had been considered. Finally, a comparison is also provided between the current work and other previous structures described in the literature to monitor the quality of the encrypted images.

The rest of this paper is organized as follows. Section 2 discusses the research related works about various encryption methods. The preliminaries including description of BBS generator and hash function are presented in Sect. 3. Sections 4, 5 and 6 describe the proposed scheme of key-streams generator, the experimental setup and NIST test results, respectively. The proposed COVID-19 image encryption scheme is given in Sect. 7 with security analysis and evaluation metrics. Comprehensive results and comparative evaluations are presented in Sect. 8. Finally, this study is concluded in Sect. 9.

2 Related Works

The strength of BBS generator has been used in various applications such as authentication and encryption tasks. Vybornova et al. [11] proposed a key-based derivation method for protecting the password using BBS generator. Thus, the algorithm is obtained for extracting cryptographic keys from passwords that are easy to recall. Generating a unique key with DNA and the generator of random numbers has been investigated in [12]. Every random sequence generated by BBS random number generator is the product of a seed value that makes the sequence unique and reproducible. A novel encryption scheme is built in [13] for securing message transfer. This scheme was implemented using BBS generator based on stream cipher algorithm. A sensor-based architecture was also proposed to achieve robustness versus channel attacks based on the scan [14]. Its design used a safe SHA-256 and BBS pseudo-random generator to create a simple challenge/response scheme. Paul et al. [15] introduced two PRNS methods including BBS, Xorshift, and permuted congruential PRNGs. The first PRNG method is generic and can be used for any application, but the second method is recommended for Internet of Things (IoT) applications with low power. In [16], a new BBS-ECPRNG method is proposed, based on the product of two elliptic curve points and the extracted number of bits per iteration. A study on new combination of advanced encryption standard with smart BBS-PRNGs is presented in [17]. It presents a hybrid metaheuristic technique of BBS and iterated local search (ILS) to generate strong crypto key using nonparametric statistical tests. Hardware-based realization of two 16-bit PN sequence generators and linear feedback shift register (LFSR) with BBS has been done using Field Programmable Gate Array (FPGA) as a practical application of communication systems [18]. Using the obtained data in [19], the Beaufort cipher key-based BBS generator was applied to protect text files. A new seeding technique is proposed in [20] which leverages sensor data to provide

safe seeds for RNG. Given the increasing proliferation of sensors and Internet access on smart devices, this technique could improve cybersecurity across several domains without additional cost. Olsson et al. [21] concluded that the GPU-based implementation cannot be a good alternative for the CPU-based implantation of the BBS, because the resulted performance of their GPU-bound BBS implementation was approximately four times slower than a CPU version with open-source GNU library for multiple precision arithmetic operations. The above problem of bit predictor for the BBS generator to the composite quadratic residue problem, which is assumed to be difficult, has been solved in [22]. A novel hardware is proposed in [23] to allow highly efficient BBS generation using Montgomery multipliers. The multiplicative offset is incorporated into cost-free BBS generation, and further simplification is made possible by considering BBS-like sequences being generated. Many architectures of the BBS generator have been analyzed with a modulo of 160 and 512-bits [24]. The use of iterative Montgomery architecture was proposed for restricted devices such as RFID tags or nodes in sensor networks. The study in [25] summarized that the BBS would contribute to the technique of encryption with RC4 to ensure the ciphertext more secure and robust. The BBS architecture of 1024-bit is constructed in [26] using modified radix-2 Montgomery modular multiplier to generate pseudo-random bit sequences. The substantial improvement of overall latency in the built BBS architecture provides secure hardware resources for various cryptography applications, such as constrained IoT devices. Shparlinski [27] proved that if the length of Rivest–Shamir–Adleman (RSA) and BBS generators is sufficiently large then the elements of such sequences are uniformly distributed modulo m with a positive proportion of the rightmost bits. A new cryptosystem is implemented in [28] based on modifications of the BBS PRNG. This study suggested that both the cubic modification and the original BBS generator have the same security performance.

3 Preliminaries

The BBS pseudo-random generator of numbers which proposed by Blum et al. [29] is described in Sect. 3.1. In addition, Sect. 3.2 addresses the mathematical cryptographic hash function, which acts as an important information security building block and is used in various security applications and protocols [30].

3.1 BBS Algorithm

The BBS is one of important cryptographic generators and also fast enough. It can be applied at least in certain special cases where the speed of computation is not highly priority. The BBS generator uses a modulus $N = p * q$ with p and q large primes [31] and is based on the hardness of quadratic problem. This problem is thought to be computationally challenging and intractable, since if the N factorization is uncertain, there does not seem to be any way to solve it effectively at the present time. The BBS structure is given by Eq. (1).

$$x_i \equiv x_{i-1}^2 \bmod N \tag{1}$$

where N is the product of two large primes, p and q, that have the following special properties in Eq. (2).

$$p \equiv 3 \bmod 4 \tag{2}$$

The initial value x_0 is a quadratic residue of N, meaning there is an integer z such that $z^2 \equiv x_0 \bmod N$. In this way, x_0 can be chosen by taking a random seed s that is satisfies $\gcd(s, N) = 1$, then $x_0 = s^2$ acts as an initialization vector.

3.2 Hash Function

A hash function H is considered a useful type of transformation, which takes X input value of any length and returns a fixed length output as shown in Fig. 1, this is commonly referred to as the hash value h ($h = H(X)$) [32]. A successful generator mechanism can be based on a hash function that is non-invertible. Here, the hash-based BBS mechanism is intended to allow the usage of any acceptable secure hashing function. Then, it can be utilized by many applications that need different security strengths. As a result, $H(X)$ calculation is a fast operation for any hash function H with input data X. So, it is computationally impossible to reverse the hash value h; i.e., a hash function should work as a one-way function. Moreover, the security usage of the hash function should be at the same level or higher than the security strength of targeted applications.

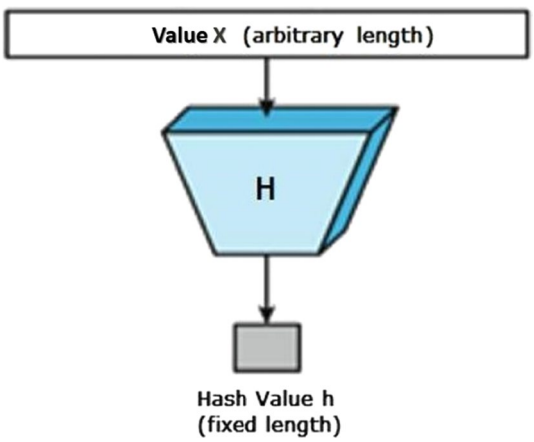

Fig. 1 Diagram of cryptographic hash function

4 Proposed Hash-BBS Generator

In order to overcome the shortcomings of BBS generator in previous studies, the proposed Hash-BBS generator implements the intractability of hash functions and integer factorization problem (IFP) into one algorithm named HBBS. The HBBS key-streams is based on two hard problems. First, the BBS generator is based on computationally challenging and intractable "quadratic problem" with large primes. Second, the secure cryptographic hash function H which has three properties: collision resistance, pre-image resistance, and second pre-image resistance. Furthermore, the encryption process based on three secure methods: both of transpose operation and S-box substitution for achieving confusion and bit permutation for diffusion. Finally, the used key space is about 2^{512} possible keys which consider sufficient to withstand channel attacks.

4.1 HBBS Generator Algorithm

The HBBS generator security is based mainly on the assumption of intractable factoring large composites. For each step, we extract some of the information from $H(X_i)$, which is typically the least significant bit. The steps of the HBBS keys generator are described in Algorithm 1.

Algorithm 1 HBBS Generator

1. Select two n-bit random primes p and q that satisfy 3 modulo 4,
2. Let $N = p * q$, and choose $0 < S < N$ to be a random integer that is relatively prime to N,
3. Compute the initial value from the seed S by the equation:

$$X_0 \equiv S^2 \ mod \ N \tag{3}$$

4. Compute the X values

$$X_i \equiv X_{i-1}^2 \ mod \ N \tag{4}$$

5. Compute the hash value h of X_i

$$h_i \equiv H(X_i) \quad i = 1,2, \ldots \tag{5}$$

6. The i^{th} pseudo-random bit b_i of HBBS is the LSB of h_i, extracting multiple keys,
   (a) $Key-1$ for **1-bit** LSB
   (b) $Key-2$ for **2-bit** LSB
   (c) $Key-3$ for **3-bit** LSB

The extracted key-streams depend on the p and q selected Blum prime numbers and the hash function H. This hash function is mainly used in key generation of random bits, so that the key bit sequence is not possible to reverse to make it cryptographically secure. Consequently, the generated key-streams are used for encryption process. As a security requirement, a hash value should not be used as an image for two distinct input messages and the input message from a given hash value should be difficult to identify.

The algorithm starts by taking two sufficiently large prime integers p and q, each has bit-length which considered sufficiently secure. Then, calculating N which is the product of p and q and select a random seed S integer value from the interval $[1, N-1]$, such that $\gcd(S, N) = 1$. Thereafter, the initial value from the seed S is computed by Eq. (3). The main X values and h values from ith iterations of steps 4 and 5 are then calculated. In the last step 6, the pseudo-random bits are extracted from the LSB of computed h values according to the number of extracted bits needed in each case. We consider three extraction cases of 1-bit ($Key-1$), 2-bit ($Key-2$) and 3-bit ($Key-3$) acting as key-streams for the next encryption scheme.

The HBBS generator's security depends on the difficulty of factoring large number $N = p*q$ [33] and the security properties of hash functions. For a given sequence obtained by the HBBS generator, the cryptanalyst cannot predict the next bit or the previous bit of the sequence (i.e., polynomial time unpredictable) [34]. Approved hash function is designed to prevent the ability to reverse the checksums that generate back to the original texts. In other words, the data can be feeding into a hashing algorithm and getting a unique string back. For well-designed cryptographic hash function, identifying a message with a given hash value is not feasible (pre-image resistance). Moreover, it is not possible to find two messages producing the same hash value (collision resistance). Furthermore, all accepted hash functions are treated as cryptographic functions [35].

5 Experimental Setup

Combining the two mentioned hard problems together presents the security of our proposed HBBS generator. Therefore, it is the best choice for accomplishing security of pseudo-random sequences of bits. In this section, we describe the implementation of the HBBS algorithm based on the programming environment of MATLAB R2019b. In this study, two prime number values are $p = 7603$ and $q = 7487$ with the random seed value $S = 7817$. Hereafter, this is used to calculate the product $N = p * q = 56923661$ and the initialization value $z_0 = S^2 \bmod N = 4181828$ such that z_0 is the secret key. SHA-256 hash algorithm is used for the purpose of integrity of the extracted binary bits.

6 NIST-Test-Suite of HBBS Key-Streams

The National Institute of Standards and Technology (NIST) [36] tests are a statistical package that consists of 16 tests. It is built to test the randomness of binary strings obtained from cryptographic pseudo-random and random number generators based on either hardware or software. The intended 16 tests concentrate on a number of diverse forms of non-randomness that may occur in binary strings. The probability value (p value) is measured by α meaning amount which is the threshold between the area of rejection and non-rejection. The significant level equals 0.01 in NIST. The sequence is not random and rejected for p value less than 0.01. In contrast, the sequence is random and passed for p value greater than 0.01.

All generated key-streams with binary sequences of the length 10^6-bits are tested using the NIST-test-suite as illustrated in Table 1. The proposed mechanism produces a good random bit-strings with high p values. The test results showed good randomness properties of three key-streams, namely $Key-1$, $Key-2$ and $Key-3$.

Table 1 Test results for HBBS key-streams

	p Value		
Test name	$Key-1$	$Key-2$	$Key-3$
Block frequency	0.03152	0.58426	0.01354
Frequency	0.03436	0.19873	0.13667
Cusum (forward)	0.05146	0.21850	0.09684
Cusum (reverse)	0.06638	0.22198	0.23864
Long runs	0.07498	0.01901	0.72171
Spectral DFT	0.09425	0.48040	0.87612
Rank	0.55079	0.69778	0.10183
L. ziv complexity	1.00000	1.00000	1.00000
Overlapping temp	0.11539	0.33548	0.86869
Nonoverlapping temp	0.11247	0.53221	0.53495
App. entropy	0.02173	0.35717	0.76483
Universal	0.51651	0.33109	0.94665
Random excursions	0.50242	0.51477	0.91588
Serial ($m=16$)	0.01143	0.13095	0.26331
Random E. variant	0.22143	0.46757	0.99557
Runs	0.90627	0.69726	0.99204
Linear complexity	0.68884	0.02270	0.54677

7 COVID-19 Image Encryption

Recently, many methods for encrypting classical images [37–39] and medical images [9, 40, 41] have been proposed in the literature. Due to the massive data and high correlation between adjacent pixels, encryption of CT-images can be a significant application such that stream cipher over block cipher is strongly favored. The key-stream must satisfy good randomness properties and high periodicity to encrypt medical images. Consequently, secure transmission of medical data is verified.

In this paper, the key-stream bit-strings generated by the proposed HBBS are used for the encryption of four COVID-19 images of chest CT scan namely "Front," "Lung," "Side" and "Top." These images are randomly selected from the public dataset [42]. This four plainimages are of size 256×256 in which each pixel value has an 8-bit and represents a number between 0 and 255. The entropy values of the four CT plainimages are illustrated in Table 2. Our encryption technique is designed to achieve a high level of securely cryptographic data. The block-diagram of the proposed encryption method is composed of four main processes as depicted in Fig. 2. The proposed scheme is started by dividing each plainimage block into 8×8 sub-blocks. Transpose operation is then applied for each sub-block matrix. It is followed by S-box substitution operation, and the bit permutation is finally achieved. The HBBS key-stream in turn is divided into 8×8 sub-keys for the purpose of encryption process. Each sub-block of the divided plainimage is XORed with the corresponding sub-key. The obtained cipher blocks are then grouped together to get the final cipherimage. The performance of the proposed encryption scheme will be assessed by the following security analysis.

7.1 Image Histogram

It is necessary to ensure that cipherimage has no statistical similarities to plainimage [43], avoiding the leakage of

Table 2 Entropy of four CT plainimages

Plainimage	Front	Lung	Side	Top
Entropy	7.4716	4.6334	6.9780	7.2465

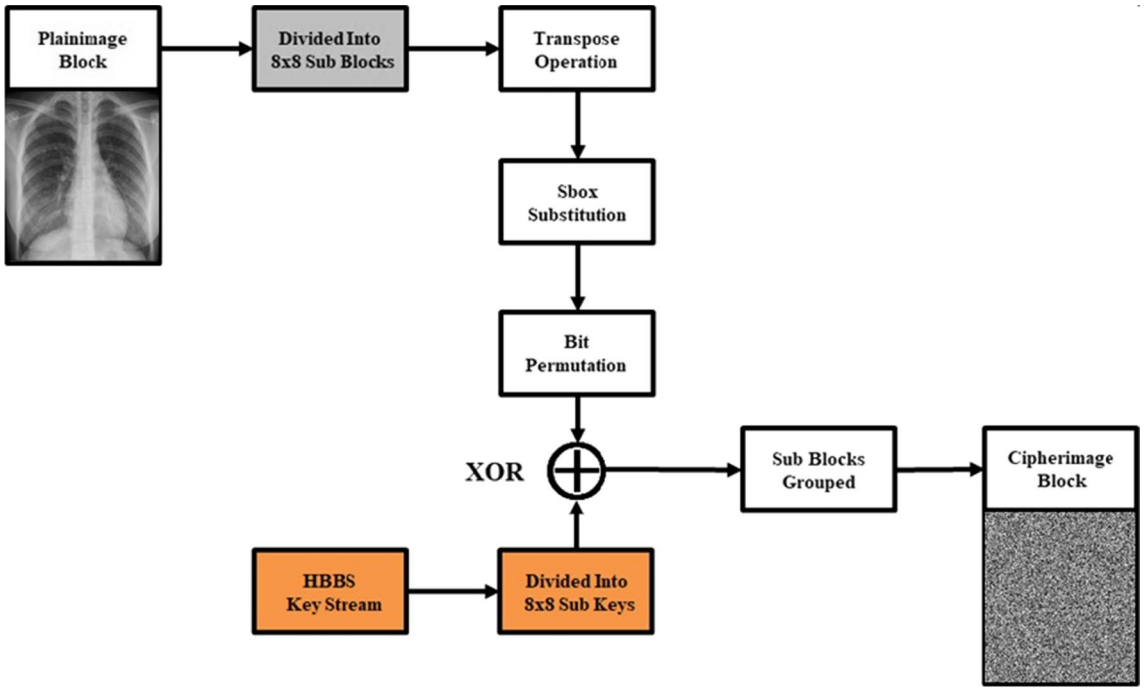

Fig. 2 Proposed scheme of HBBS-based COVID-19 CT-image encryption

information to the opponent. As shown in Fig. 3, four plainimages and the corresponding ciphered images histograms are plotted. Each plainimage histogram has large spikes, whereas the histograms of their cipher images with the three key-streams are almost smooth and uniform, indicating that each pixel is equally likely to occur. The flat cipherimages differ greatly from the respective histograms of the plainimage. Therefore, they have no proof that any statistical attack may be vulnerable to the proposed encryption scheme.

7.2 Entropy

Entropy of an image ENT(x) is a perfect measure of the randomness of that image pixels [44]. It can be calculated as

$$\mathrm{ENT}(x) = -\sum_{i=0}^{255} P(x_i) log_2 P(x_i) \tag{6}$$

where, $P(x_i)$ is the probability of x_i. Table 3 indicates values of the entropy metric for the plain and encrypted images considering by the used three key-streams as shown in Fig. 3. These entropy values are very similar to the theoretical value = 8, indicating that all encrypted image pixels are just as likely to occur. Consequently, the data leakage of the cipherimages is negligible and secured from the entropy-based attack.

7.3 Peak Signal-to-Noise Ratio (PSNR)

In the field of image processing, PSNR is mainly used as a standard image quality metric [45]. High values of the PSNR represent a good quality of the tested images. The efficiency of cipherimages with key-stream encryption is determined based on the PSNR metric and the measurement values are illustrated in Table 3. Obviously, the results showed that the proposed HBBS method is perfectly suitable for many types of medical image encryption schemes.

7.4 Structural Similarity (SSIM)

The SSIM index is used by measuring the relation between the two images to predict the perceived consistence [46]. It is a measure of comparison that calculates the image quality without distortion as the reference, based on an original uncompressed image. As illustrated in Table 3, all SSIM values are very near to zero value, indicating that no structural similarity between any of the plainimages and their corresponding cipherimages.

7.5 Mean Absolute Error and Mean Square Error

A substantial difference must be shown by the cipherimage with its corresponding plainimage. Two key techniques, mean absolute error (MAE) and mean square error (MSE) [43], are able to accomplish this task. Different values of MAE and MSE can be computed by the following equations:

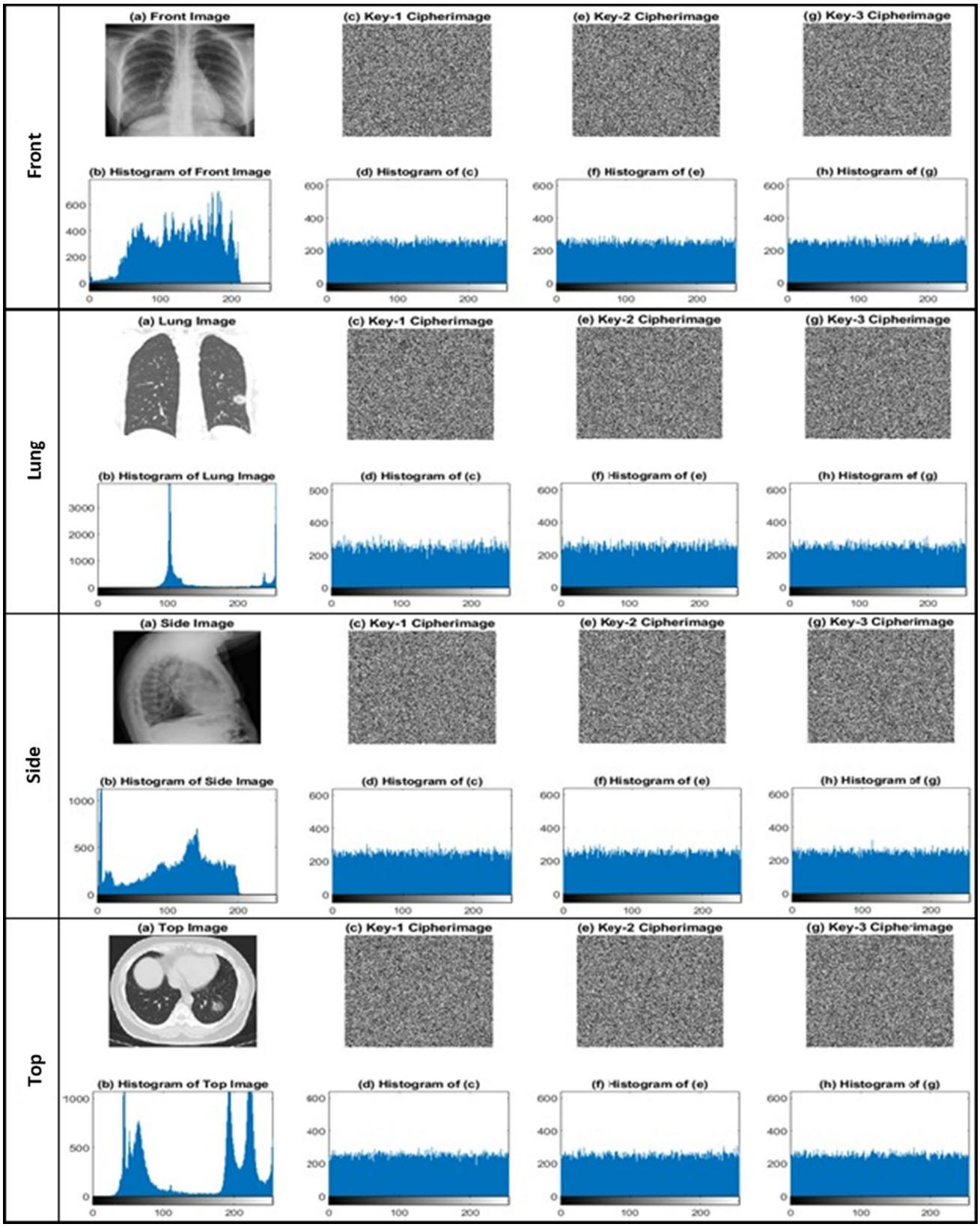

Fig. 3 Histograms of four CT plainimages and its corresponding cipherimages with three key-streams namely $Key-1$, $Key-2$ and $Key-3$

Table 3 Entropy and basic parameters for four CT cipherimages

Scheme	Entropy	PSNR	SSIM	MAE	MSE	NPCR	UACI
	Front image						
$Key-1$	7.9973	9.2311	0.0104	73.21	7822.86	99.63	28.71
$Key-2$	7.9973	9.2049	0.0106	73.43	7870.22	99.59	28.80
$Key-3$	7.9970	9.2038	0.0087	73.53	7872.32	99.60	28.84
	Lung image						
$Key-1$	7.9938	6.8536	0.0120	95.49	13,524.32	99.62	37.45
$Key-2$	7.9949	6.9029	0.0084	94.91	13,371.77	99.61	37.22
$Key-3$	7.9961	6.8987	0.0099	94.90	13,384.70	99.65	37.22
	Side image						
$Key-1$	7.9971	8.3031	0.0089	80.30	9686.65	99.61	31.49
$Key-2$	7.9970	8.3129	0.0088	80.43	9664.83	99.67	31.54
$Key-3$	7.9975	8.2868	0.0086	80.64	9723.08	99.60	31.63
	Top image						
$Key-1$	7.9974	7.4692	0.0099	88.53	11,737.12	99.60	34.72
$Key-2$	7.9971	7.4312	0.0092	88.95	11,840.15	99.61	34.88
$Key-3$	7.9976	7.4474	0.0071	88.74	11,796.21	99.61	34.80

$$\mathrm{MAE} = \frac{1}{W * H} \sum_{j=1}^{H} \sum_{i=1}^{W} \left| \left(P_{ij} - C_{ij} \right) \right| \tag{7}$$

$$\mathrm{MSE} = \frac{1}{W * H} \sum_{j=1}^{H} \sum_{i=1}^{W} \left(P_{ij} - C_{ij} \right)^2 \tag{8}$$

where W and H parameters are the width and height of the image. The gray level of the pixel in the plainimage is represented as P_{ij}, and the gray level of the pixel in the cipherimage is C_{ij}. Table 3 records the MAE and MSE values of the obtained cipherimages. High values of MAE and MSE tests verified that our encryption method is being highly resistant to the types of attacks for sensitive medical images.

7.6 Diffusion Analysis (DA)

Two specific measurements are used to quantify diffusion and ambiguity requirements: number of pixel change rate (NPCR) and unified average change intensity (UACI) [47]. The average NPCR and UACI values are listed in Table 3, with only one-bit difference in each of the four plainimages. The results showed that the average percentage values of pixels changed in encrypted image are greater than 99.61% for NPCR and 33.09% for UACI for all key-streams generated and thus better resistance to differential attacks.

Table 4 Correlation coefficients for four CT plainimages and cipherimages

Scheme	Front image			Lung image		
	Horizontal	Vertical	Diagonal	Horizontal	Vertical	Diagonal
Plainimage	0.99350	0.99523	0.98994	0.98323	0.99083	0.97650
$Key-1$	0.00029	0.00143	0.00193	0.00121	0.00520	− 0.00489
$Key-2$	0.00222	0.00190	− 0.00275	0.00360	− 0.00058	− 0.00363
$Key-3$	0.00315	0.00280	0.00503	− 0.00178	0.00295	− 0.00016
Scheme	Side image			Top image		
	Horizontal	Vertical	Diagonal	Horizontal	Vertical	Diagonal
Plainimage	0.99683	0.99659	0.99399	0.95738	0.96413	0.93398
$Key-1$	− 0.00029	− 0.00205	− 0.00107	0.00186	0.00480	− 0.00723
$Key-2$	− 0.00072	− 0.00662	0.00181	0.00678	0.00456	0.00509
$Key-3$	0.00172	0.00338	− 0.00312	0.00348	0.00368	0.00044

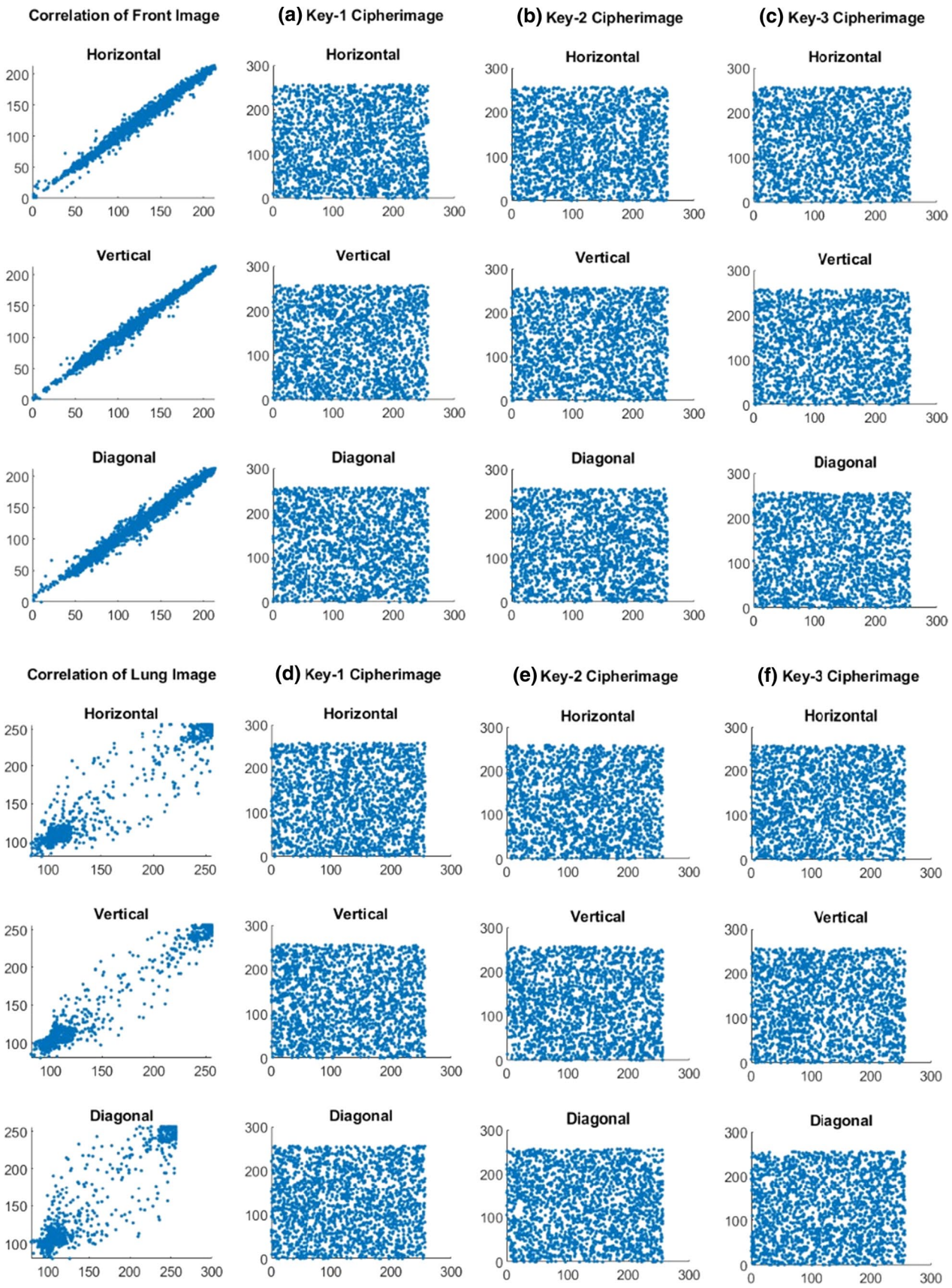

Fig. 4 Correlation coefficients of Front and Lung plainimages and its corresponding cipherimages with HBBS scheme

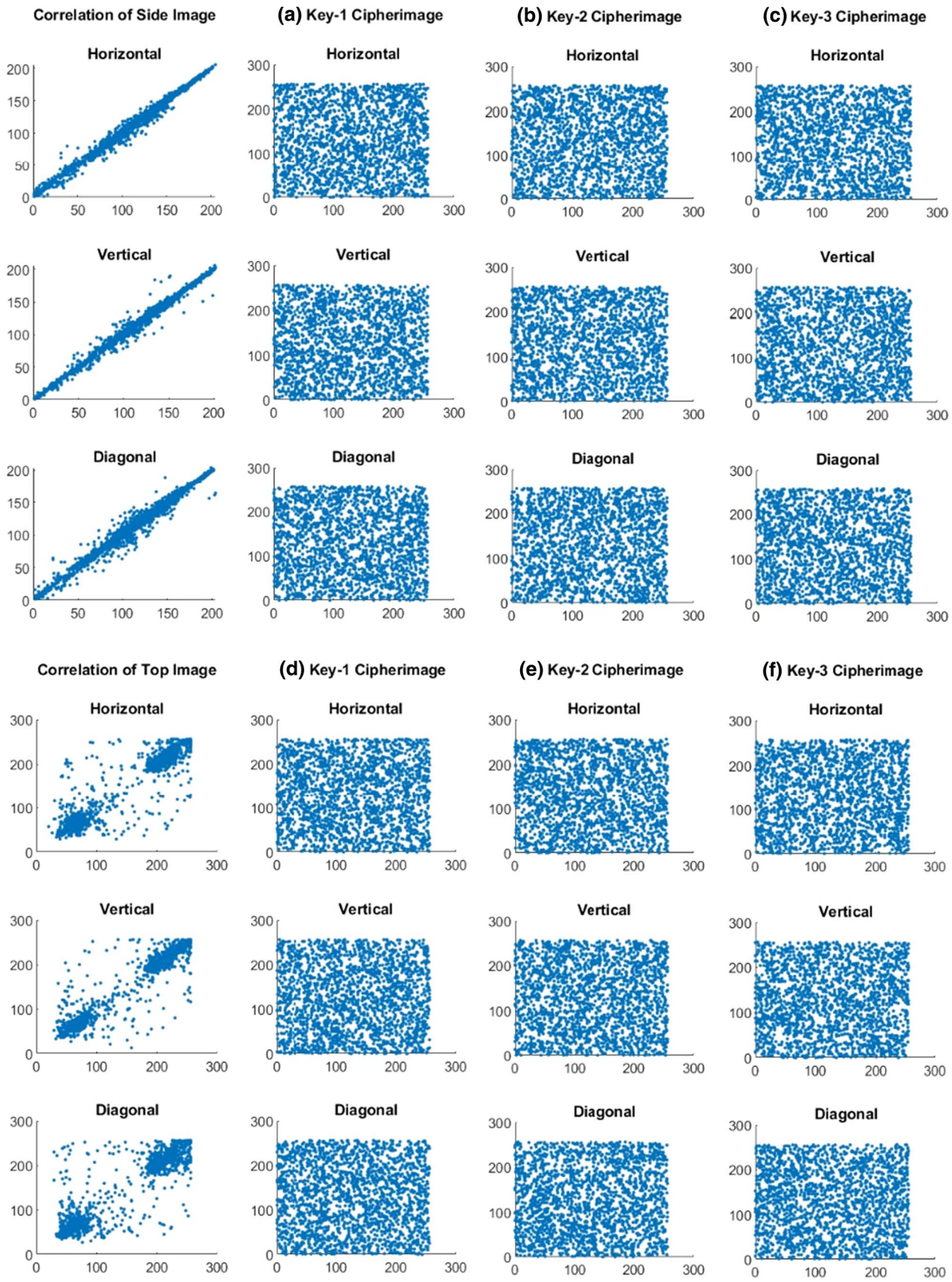

Fig. 5 Correlation coefficients of Side and Top plainimages and its corresponding cipherimages with HBBS scheme

7.7 Correlation Coefficient Measure

For any particular image, two adjacent pixels are highly correlated vertically, horizontally and diagonally within a plainimage. Maximum correlation coefficient value is one and the minimum value is zero [43]. The correlation coefficient values are obtained as listed in Table 4 for four plainimages and cipherimages, respectively. Moreover, Figs. 4 and 5 depict that a minimal correlation is appeared in the cipherimage between the adjacent pixels, even though these pixels are strongly correlated in the plainimage.

7.8 Security of the Key Space

The key space contains a set of all possible keys that may be involved in the encryption process. In [48], the effective key length is required to be more than 2^{100} possible keys. In our proposed HBBS method, the initial values are obtained with two primes p and q of at least 128-bits each and SHA-256 hash function. Then, the key space containing $2^{128} \times 2^{128} \times 2^{256} = 2^{512} = 1.3407 \times 10^{154}$ possible keys. The total key length in that case is greater than 2^{100}, which is convenient to withstand an exhaustive search attack.

8 Comparative Performance Evaluation

This section provides a comparison of our proposed method with previously implemented methods in the literature [40, 41, 49, 50]. The performance of our proposed method has been evaluated by conducting several tests based on image quality and other assessment criteria. Comparison of PSNR, SSIM, Entropy, MAE, MSE, NPCR and UACI are listed in Table 5, while Table 6 shows the comparison of the correlation coefficient values with other previous methods. Obviously, the proposed HBBS scheme outperforms other encryption techniques, representing a secure alternative to encrypt and decrypt medical images especially CT-images of COVID-19 infections.

Table 6 Comparative coefficients of correlation with other methods in the literature

Scheme	Horizontal	Vertical	Diagonal
Ref. [40]	− 0.0057	0.0090	0.0100
Ref. [41]	0.0403	0.0366	–
Ref. [49]	0.0035	0.0002	− 0.0013
Ref. [50]	0.0025	0.0062	0.0031
Key − 1	0.0019	0.0048	− 0.0072
Key − 2	0.0068	0.0046	0.0051
Key − 3	0.0035	0.0037	0.0004

9 Conclusions and Future Work

This study suggested a new mechanism to generate pseudo-random bit-strings to achieve high levels of security based on the HBBS generator and hash function. The proposed method is provided to encrypt COVID-19 CT-images into cipherimages for secured real-world transmission. In particular, the SHA-256 hash algorithm is used for the integrity and randomness of the extracted pseudo-random bits for the generation of multiple key-streams. The results of performance evaluation showed that our proposed approach has high security and good efficiency to be suitable for smart health applications and telemedicine.

In the future work, optimizing time processing of CT-image encryption using advanced techniques of artificial intelligence (AI) will be considered to satisfy requirements of real-time communications.

Table 5 Comparative several measures with other methods in previous studies

Scheme	Entropy	PSNR	SSIM	MAE	MSE	NPCR	UACI
Ref. [40]	7.9974	53.8291	–	–	–	99.58	33.42
Ref. [41]	7.6850	8.4886	0.0075	–	9209.21	–	–
Ref. [49]	7.9987	–	–	81.21	–	75.50	31.81
Ref. [50]	7.9961	–	–	–	–	99.63	33.43
Key − 1	7.9974	7.4692	0.0099	88.53	11,737.12	99.60	34.72
Key − 2	7.9973	9.2049	0.0106	73.43	7870.22	99.59	28.80
Key − 3	7.9976	7.4474	0.0071	88.74	11,796.21	99.61	34.80

Acknowledgments The authors would like to thank the editors and anonymous reviewers for their valuable comments and suggestions to enhance the readability of this manuscript.

Compliance with Ethical Standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Paules, C.I.; Marston, H.D.; Fauci, A.S.: Coronavirus infections more than just the common cold. JAMA **323**(8), 707–708 (2020)
2. Li, Q., et al.: Early transmission dynamics in Wuhan, China, of novel coronavirus infected pneumonia. N. Engl. J. Med. **382**, 1199–1207 (2020)
3. Reyad, O.: Novel coronavirus COVID-19 strike on Arab countries and territories: a situation report I. arXiv:2003.09501 (2020)
4. World Health Organization (WHO) Coronavirus disease (COVID-19): weekly operational update on COVID-19 (2020). https://covid19.who.int/. Accessed 23 Oct 2020
5. Karar, M.E.; Hemdan, E.E.; Shouman, M.A.: Cascaded deep learning classifiers for computer-aided diagnosis of COVID-19 and pneumonia diseases in X-ray scans. Complex Intell. Syst. (2020). https://doi.org/10.1007/s40747-020-00199-4
6. Liu, H., et al.: Clinical and CT imaging features of the COVID-19 pneumonia: focus on pregnant women and children. J. Infect. **80**(5), e7–e13 (2020)
7. Zeineldin, R.A.; Karar, M.E.; Coburger, J., et al.: DeepSeg: deep neural network framework for automatic brain tumor segmentation using magnetic resonance FLAIR images. Int J CARS **15**, 909–920 (2020)
8. Zhang, F.Y.; Qiao, Y.; Zhang, H.: CT imaging of the COVID-19. J. Formos. Med. Assoc. **119**(5), 990–992 (2020)
9. Reyad, O.; Hamed, K.; Karar, M.E.: Hash-enhanced elliptic curve bit-string generator for medical image encryption. J Intell Fuzzy Syst **39**(5), 7795–7806 (2020)
10. Pavithra, V.; Jeyamala, C.: A survey on the techniques of medical image encryption. In: IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), pp. 1–8 (2018)
11. Vybornova, Y.D.: Password-based key derivation function as one of Blum–Blum–Shub pseudorandom generator applications. Proced. Eng. **201**, 428–435 (2017)
12. Sodhi, G.K.; Gaba, G.S.: DNA and Blum Blum Shub random number generator based security key generation algorithm. Int. J. Secur. Appl. **11**(4), 1–10 (2017)
13. Assa, B.; Khaled, M.; Lakhdar, G.: Implementation of Blum Blum Shub generator for message encryption. In: International Conference on Control, Engineering and Information Technology (CEIT14), Proceedings IPCO, pp. 118–123 (2014)
14. Koopahi, E.; Borujeni, S.E.: Secure scan-based design using Blum Blum Shub algorithm. In: IEEE East-West Design and Test Symposium (EWDTS), Yerevan, pp. 1–5 (2016)
15. Paul, B., et al.: Design and implementation of low-power high-throughput PRNGs for security applications. In: 32nd International Conference on VLSI Design and 18th International Conference on Embedded Systems (VLSID), pp. 535–536 (2019)
16. Omorog, C.D.; Gerardo, B.D.; Medina, R.P.: Enhanced pseudorandom number generator based on Blum–Blum–Shub and elliptic curves. In: IEEE Symposium on Computer Applications and Industrial Electronics (ISCAIE), pp. 269–274 (2018)
17. Kadhim, E.A.; Hussein, Z.K.; Hadi, H.J.: AES cryptography algorithm based on intelligent Blum–Blum–Shub PRNGs. J. Eng. Appl. Sci. **12**, 9035–9040 (2017)
18. Sewak, K.; Rajput, P.; Panda, A.K.: FPGA implementation of 16 bit BBS and LFSR PN sequence generator: a comparative study. In: IEEE Students' Conference on Electrical, Electronics and Computer Science, Bhopal, pp. 1–3 (2012)
19. Sari, R.N.; Hayati, R.S.: Beaufort cipher algorithm analysis based on the power lock-Blum Blum Shub in securing data. In: 6th International Conference on Cyber and IT Service Management (CITSM), Indonesia, pp. 1–4 (2018)
20. Hong, S.L.; Liu, C.: Sensor-based random number generator seeding. IEEE Access **3**, 562–568 (2015)
21. Olsson, M.; Gullberg, N.: Blum Blum Shub on the GPU. Master Thesis, Sweden (2012)
22. Klein, A.: The Blum-Blum-Shub generator and related ciphers. In: Klein, A. (ed.) Stream Ciphers, pp. 241–257. Springer, London (2013)
23. Parker, M.G.; Kemp, A.H.; Shepherd, S.J.: Fast BBS-sequence generation using Montgomery multiplication. IEE Proc.—Comput. Digit. Tech. **147**(4), 252–254 (2000)
24. Lopez, P.P., et al.: Cryptographically secure pseudo-random bit generator for RFID tags. In: International Conference for Internet Technology and Secured Transactions, London, IEEE, pp. 1–6 (2010)
25. Siahaan, A.P.: Blum Blum Shub in generating key in RC4. Int. J. Sci. Technol. **4**(10), 1–5 (2016)
26. Panda, A.K.; Ray, K.C.: Design and FPGA prototype of 1024-bit Blum–Blum–Shub PRBG architecture. In: IEEE International Conference on Information Communication and Signal Processing (ICICSP), Singapore, pp. 38–43 (2018)
27. Shparlinski, I.: RSA and Blum–Blum–Shub generators of pseudo-random numbers. Number Theor. Methods Cryptogr., Prog. Comput. Sci. Appl. Log. **17**, 131–141 (1999)
28. Knuth, T.: Nonquadratic Variation of the Blum–Blum–Shub Pseudorandom Number Generator. The NPS Institutional Archive, Calhoun (2016)
29. Blum, L.; Blum, M.; Shub, M.: A simple unpredictable pseudo-random number generator. SIAM J. Comput. **15**(2), 364–383 (1986)
30. Buchmann, J.A.: Cryptographic hash functions. In: Buchmann, J.A. (ed.) Introduction to Cryptography, pp. 235–248. Springer, New York, NY (2004)
31. Schneier, B.: Applied Cryptography: Protocols, Algorithms, and Source Code in C, Wiley (2015)
32. Smart, N.P.: Hash functions, message authentication codes and key derivation functions. In: Smart, N.P. (ed.) Cryptography Made Simple, pp. 271–294. Springer International Publishing, Cham (2016)
33. Sidorenko, A.; Schoenmakers, B.: Concrete security of the Blum–Blum–Shub pseudorandom generator. In: Smart, N.P. (Ed.) Cryptography and Coding, LNCS, 3796th edn. Springer, Heidelberg (2005)
34. Junod, P.: Cryptographic secure pseudo-random bits generation: the Blum-Blum-Shub generator (1999). https://crypto.junod.info/bbs.pdf. Accessed 23 Oct 2020
35. Dang, Q.H.: Secure hash standard. No. Federal Inf. Process. Stds. (NIST:FIPS)-180-4 (2015)
36. Barker, E.B.; Kelsey, J.M.: Recommendation for Random Number Generation Using Deterministic Random Bit Generators (Revised). US Department of Commerce, Technology Administration, NIST, Computer Security Division, Information Technology Laboratory (2007)
37. Abd-Elhafiez, W.M.; Reyad, O.; Mofaddel, M.A.; Fathy, M.: Image encryption algorithm methodology based on multi-mapping

image pixel. In: Hassanien, A. et al. (eds.) AMLTA 2019, AISC vol. 921, pp. 645–655. Springer, Cham (2020)
38. Reyad, O.: Text message encoding based on elliptic curve cryptography and a mapping methodology. Inf. Sci. Lett. **7**(1), 7–11 (2018)
39. Zhang, G.; Liu, Q.: A novel image encryption method based on total shuffling scheme. J. Opt. Commun. **284**(12), 2775–2780 (2011)
40. Chai, X.; Zhang, J.; Gan, Z., et al.: Medical image encryption algorithm based on Latin square and memristive chaotic system. Multimed. Tools Appl. **78**, 35419–35453 (2019)
41. Sasikaladevi, N.; Geetha, K.; Revathi, A.: EMOTE–Multilayered encryption system for protecting medical images based on binary curve. J. King Saud. Univ-Comput. Inf. Sci. (2019). https://doi.org/10.1016/j.jksuci.2019.01.014
42. Cohen, J.P.; Morrison, P.; Dao, L.: COVID-19 image data collection. arXiv, vol. eess.IV, p. 2003.11597 (2020)
43. Gonzalez, R.C.; Woods, R.E.: Digital Image Processing, 3rd edn. Prentice-Hall Inc, Upper Saddle River (2006)
44. Shannon, C.E.: Communication theory of secrecy systems. Bell Syst. Tech. J. **28**(4), 656–715 (1949)
45. Reyad, O.; Khalifa, H.S.; Kharabsheh, R.: Image pixel permutation operation based on elliptic curve cryptography. J. Appl. Math. Inf. Sci. **13**, 183–189 (2019)
46. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
47. Wu, Y.; Noonan, J.P.; Agaian, S.: NPCR and UACI randomness tests for image encryption. Cyber J. Multidiscip. J. Sci. Technol. J. Sel. Areas Telecommun. (JSAT) **1**(4), 31–38 (2011)
48. Alvarez, G.; Li, S.: Some basic cryptographic requirements for chaos-based cryptosystems. Int. J. Bifurc. Chaos **16**(8), 2129–2151 (2006)
49. Ismail, S.M., et al.: Generalized double-humped logistic map-based medical image encryption. J. Adv. Res. **10**, 85–98 (2018)
50. Liu, J.; Ma, Y.; Li, S., et al.: A new simple chaotic system and its application in medical image encryption. Multimed. Tools Appl. **77**, 22787–22808 (2018)