Applied Acoustics 203 (2023) 109196

Contents lists available at ScienceDirect

Applied Acoustics

journal homepage: www.elsevier.com/locate/apacoust

Audio encryption using two chaotic map based dynamic diffusion and double DNA encoding

Atul Kumar *, Mohit Dua

Department of Computer Engineering, NIT, Kurukshetra, India

ARTICLE INFO

Article history:
Received 23 April 2022
Received in revised form 11 December 2022
Accepted 29 December 2022
Available online 4 January 2023

Keywords:
Security
Audio encryption
Chaos
Bifurcation
DNA
PSNR

ABSTRACT

A large amount of data in the form of audios are communicated all over the world. However, the attacker can steal this data, and then misuse this information. Hence, it is necessary to create strong audio encryption strategy to protect the data from various types of existing threats. This paper discusses about the implementation of a novel chaos-based strategy for audio encryption that uses the Sine-Cosine (SC) Map, Logistic Sine Cosine (LSC) map, and double DNA operation. The proposed approach is divided into three parts, i.e., Firstly, the permutation is performed using the Sine-Cosine map to generate the permuted audio. The permuted audio samples are scrambled using the double DNA addition encoding in the second phase. Finally, dynamic diffusion is carried out by selecting one of the chaotic maps from SC Chaotic map and the LSC map. The evaluation of the proposed approach has been done using various standard metrics like Peak Signal to Noise Ratio (PSNR), Histogram Analysis, Number of Sample Change Rate (NSCR), Correlation Coefficient (CC).

© 2022 Elsevier Ltd. All rights reserved.

1. Introduction

Internet is an enormously multiuse source that helps us to complete several tasks quickly and efficiently. It is broadly used for many different activities like education, research, communication, etc. These activities need the exchange of different types of information like audio, videos, and images. However, the attacker tries to steal this data during the communication of the information. Therefore, the encryption of data is required for hiding this information. Encryption converts the multimedia data i.e., images, audios, videos into the noise like data. In recent years, researchers have developed various chaos based techniques for the encrypting audio data using the Deoxyribonucleic acid (DNA) encoding [1–3], permutation and substitution architecture [4,5], and cellular automata [6].

In the last two decades, chaos theory has been used for developing various encryption strategies for the multimedia applications. The chaotic map provides the nonlinearity and is sensitive to the initial condition [7,8]. Many researchers have proposed different chaotic maps for encrypting different forms of data [9–11]. As per the chaotic equation used, the chaotic maps are categorised as one dimensional chaotic map [12–14], two-dimensional chaotic map [15–17], three-dimensional chaotic map [18–20], fourth dimensional chaotic [21–23], fifth dimensional chaotic system [24–26], sixth dimensional system [27,28] etc. The Logistic map (LM) [29], Sine map(SM) [30] and Tent map (TM) [31] are the traditional chaotic maps. These chaotic maps are simple to implement. However, the traditional chaotic maps contain some of the security issues like blank window, stable window and small chaotic range [32,33]. The multidimensional chaotic maps are more secure and have larger chaotic range than the single dimensional chaotic system, however, multidimensional chaotic maps are complex to implement. Hence, researchers use one dimensional hybrid maps to overcome all the issues of traditional and multidimensional chaotic system. Further, image cryptosystems based on DNA methods have recently become more popular. These cryptosystem use DNA as information carrier. High parallelism, huge data storage, and an extended security are the key characteristics of DNA-based computation [34–38].

This paper also uses two one-dimensional hybrid chaotic maps, that are Sine-Cosine chaotic map (SC) and Logistic Sine Cosine (LSC) map. Also, the double DNA addition operation is used for the implementation of proposed audio cryptosystem. The rest of the paper is organized as: Section 2 discusses the existing related work for the audio encryption. The basics DNA encoding and DNA addition operations are given in section 3. Section 4 describes the fundamentals of SC Chaotic map and LSC map, the architecture of the proposed scheme is explained in Section 5, Section 6 presents the experimental set up and result analysis, and Section 7 gives the conclusion and future work.

* Corresponding author.
E-mail addresses: atul.kumar1995@gmail.com (A. Kumar), er.mohitdua@nitkkr.ac.in (M. Dua).

https://doi.org/10.1016/j.apacoust.2022.109196
0003-682X/© 2022 Elsevier Ltd. All rights reserved.

2. Related work

In the last two decades, various chaos based encryption algorithms have been developed by different researchers. However, in last few years, many researchers have worked in the area of audio encryption, also [2,6,39–43]. They have used various advanced techniques such as DNA [44,45], Cellular Automata Theory [6] etc. to extend the security provided by the chaos functions.

Yang et al. [39] developed an encryption strategy for the audio encryption using the quantum mechanism. Two secret random encoding phases, and Quantum Fourier Transform (QFT) are used for their proposed encryption strategy. Simulation findings and theoretical analysis reveal that the suggested strategy outperforms its conventional alternatives in terms of resilience towards attacks and computing complexity.

George et al. [6] used linear feedback shift register (LFSR) and two-dimensional cellular automata for implementation of a compressive sensing based audio encryption scheme. The authors used a secure measurement matrix based on linear feedback shift registers (LFSR). An LFSR-based random bit stream generator is used for generating the initial state matrix of cellular automata.

In 2017, Sathiyamurthi et al. [46] proposed speech encryption strategy using the chaotic shift keying. This scheme used four different chaotic maps such as LM, Tent map, Quadratic map and the Bernoulli map. The authors claimed that the encryption strategy provides powerful diffusion and confusion mechanism for the speech communication. In the same year, the authors of [47,48] proposed the audio encryption mechanism using mixed chaos and wavelet function, respectively.

In 2019, the Wang et al. [44] suggested a chaos based audio encryption scheme using the DNA approach. In the proposed method, the seed value of the chaotic map is initialized using the hash value of the audio. The authors used the DNA coding and the chaotic system for confusing and diffusion. In the same year, Kalpana et al. [49] and Kordov et al. [5] proposed audio encryption scheme using fuzzy cellular neural networks (FCNN) and permutation-substitution architecture, respectively.

Kaur et al. [42] designed encryption algorithm for the speech signal using four different chaotic maps such as LM, Cubic map, Quadratic map and Skew-tent map. The authors performed various test for analysing their scheme and claimed that the encryption strategy provides good results.

It is an important fact that a for an efficient and secure encryption scheme, a chaos map should be simple to implement, should have a larger chaotic range and also, should not suffer from problems of blank and stable. Motivated by this fact, this paper develops a novel encryption strategy for the audio encryption uses the combination of a SC Map, LSC map, and double DNA addition operation. The proposed encryption strategy also uses a 128-bit shared key and 128-bit initial Vector. In the first phase, the permutation operation is carried out using the SC map and shared key. The permuted audio is further dynamically diffused using the SC map, DNA addition operation, and LSC map during the second phase. This paper chooses a chaotic sequence dynamically from the given two chaotic maps i.e., SC map & LSC map. Hence, it becomes difficult for the attacker to guess the correct chaotic map, which in turn, enhances the security of the proposed scheme. Testing is carried out by various metrics like Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR), Histogram Analysis, Correlation Coefficient, Number of Sample Change Rate (NSCR).

3. Deoxyribonucleic acid (DNA)

This paper uses the DNA operation along with two different Chaotic maps i.e., SC map and LSC map. Deoxyribonucleic acid (DNA) contains the four nucleic acids A (Adenine), G (Guanine), C (cytosine), T (Thymine) [37,50]. Tables 1 and 2 represent the DNA encoding rule and DNA addition operation, respectively.

4. Chaos map

This paper uses the DNA operation along with two different Chaotic maps i.e., SC map and LSC map, where the SC map has been designed by the authors of this paper and the LSC map has been proposed by Hua et al. [51]. The SC map has been designed using the combination of two Trigonometry functions that are sine and cosine function. Mathematical equation of the SC Chaotic map is defined as:

$$x_{i+1} \leftarrow |(|sin(-mx_i + x_i^3 - m * sin(x_i))| - |cos(-mx_i + x_i^3 - m * sin(x_i))|)| \tag{1}$$

where x_0 defines as the initial seed, m defines is the control parameter, and $m \in [2.2, 8]$.

LSC map [51] is designed using the combination of the three different maps i.e., Cosine, Sine and Logistic map. The mathematical equation for the LSC map is defined in equation (2).

$$x_{i+1} = cos(\pi(4rx_i * (1 - x_i) + (1 - r) * sin(\pi x_i) - 0.5)) \tag{2}$$

where x_i defines as the ith position of the variable x, x_0 defines as the initial seed, and r refers as the control parameter, $r \in [0, 1]$.

Fig. 1 shows the bifurcation diagram comparison of SC Map and LSC Map with LM map, SM, and TM. It can be observed that the proposed SC chaos map and LSC Map have larger chaotic range, and do not suffer from the blank window and stable windows problem within the specified chaotic range.

Similarly, Fig. 2 shows the Lyapunov exponent comparison of SC Map and LSC Map with LM, SM, and Tent map. It can be observed that the proposed SC chaos map and LSC Map have more positive Lyapunov exponent within the specified chaotic range.

5. Proposed architecture

Fig. 3 displays the detail architecture of the encryption strategy. The proposed algorithm uses a *ShareKey* (secret key) and an *InitalVector* (random number) each of size 128-bit, respectively. The initial vector is further divided into four parts i.e., *I*0, *I*1, *I*2, *I*3 each of 32-bit respectively. Firstly, the seed (*IP_SC*), and control parameter (*CP_SC*) for the SC map are generated using the variable *ShareKey* and *I*2, *I*3. Similarly, the seed (*IP_LSC*) and the control parameter (*CP_LSC*) for the LSC map are created. Then, the proposed encryption scheme generates a sequence of SC map (*SineCosineSequence*) and a LSC map (*LogisticSineCosineSequence*)

Table 1
DNA encoding rules.

Rule	0	1	2	3	4	5	6	7
00	A	A	T	T	C	C	G	G
01	C	G	C	G	A	T	A	T
11	G	C	G	C	T	A	T	A
10	T	T	A	A	G	G	C	C

Table 2
DNA addition operation.

ADD	A	C	G	T
A	A	C	G	T
C	C	G	T	A
G	G	T	A	C
T	T	A	C	G

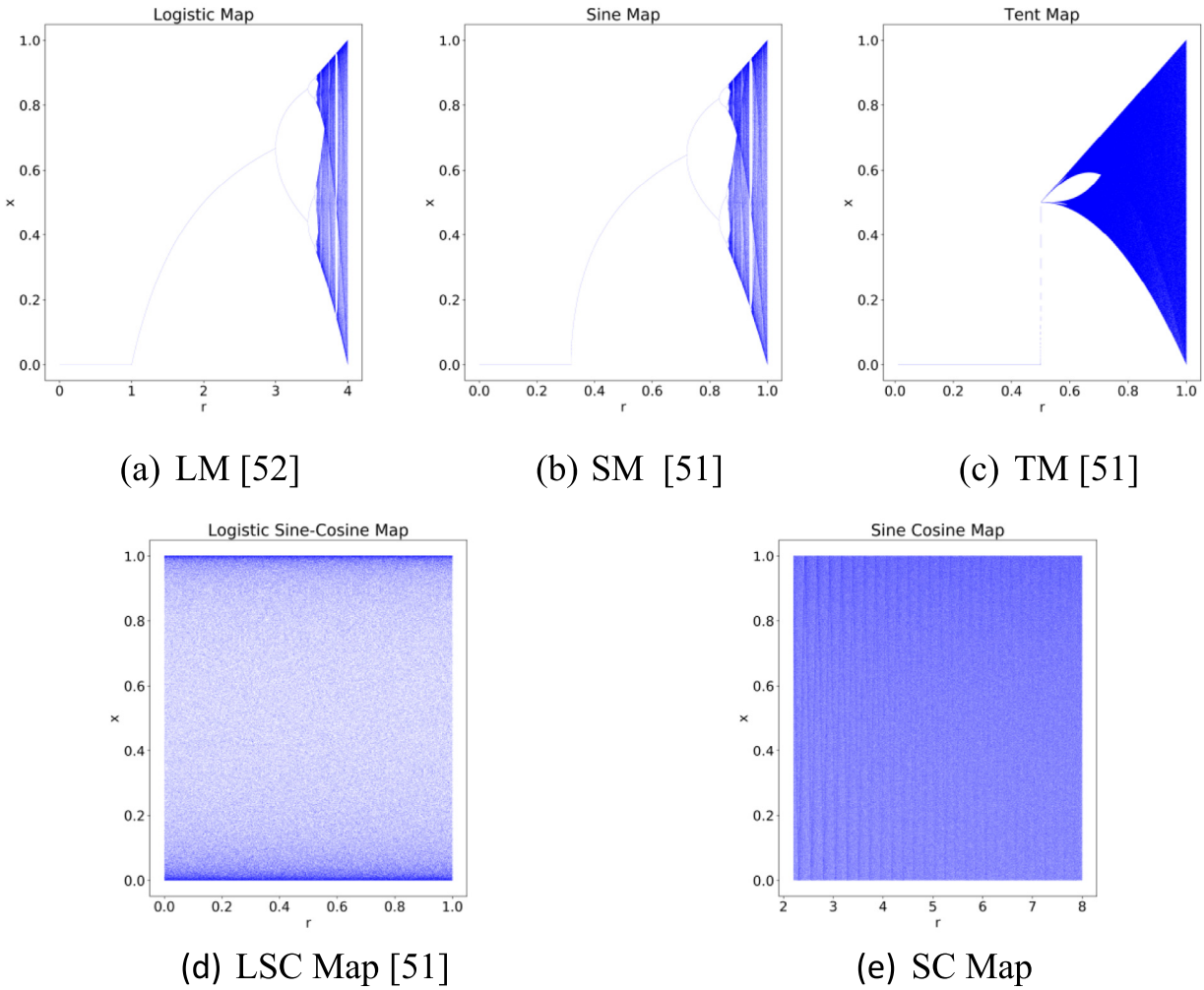


Fig. 1. Bifurcation Diagram of Chaotic Map.

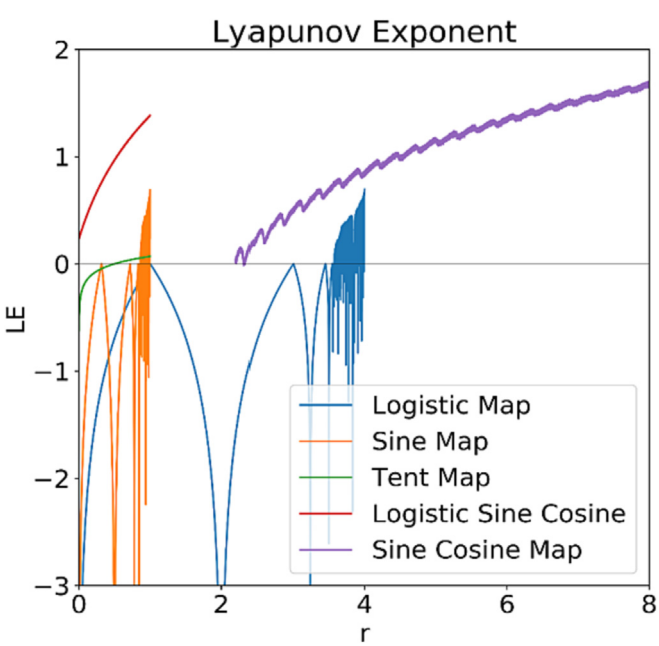


Fig. 2. Lyapunov Exponent of Different Chaotic Maps.

using the pairs (*IP_SC*,*CP_SC*) and (*IP_LSC*,*CP_LSC*), respectively. Two different pair of intermediate keys (*IntermediateKey*1, *IntermediateKey*2) are also generated using the *ShareKey* and *I*0, *I*1 vectors. In second step, the audio permutation is performed using the SC chaotic map and *I*3.

After that, the sequence of SC map and LSC map are multiplied with the *IntermediateKey*1, and *IntermediateKey*2 to create the chaotic map sequences *CM*1, and *CM*2 respectively. Then, the permutated audio, and the *CM*1, *CM*2 are converted into the 16-bit binary values. In the third step, the double DNA addition operation is applied on these binary values, and dynamic diffusion is performed selecting one of the chaotic maps. The following subsections describe these three steps in detail.

5.1. Initialization

In this phase, the control parameters and initial parameters for the Sine-Cosine Chaotic map (*IP_SC*,*CP_SC*) and Logistic Sine Cosine map (*IP_LSC*,*CP_LSC*) are created using the *ShareKey*, *InitialVector* and *I*2, *I*3 as shown by equation (3) - (6). The chaotic sequences of Sine Cosine map are created according to the equation (1) using the seed (*IP_SC*) and control parameter (*CP_SC*). Then, this sequence is stored into the *SineCosineSequence* variable. Similarly, the chaotic sequences of Logistic Sine Cosine map are generated according to

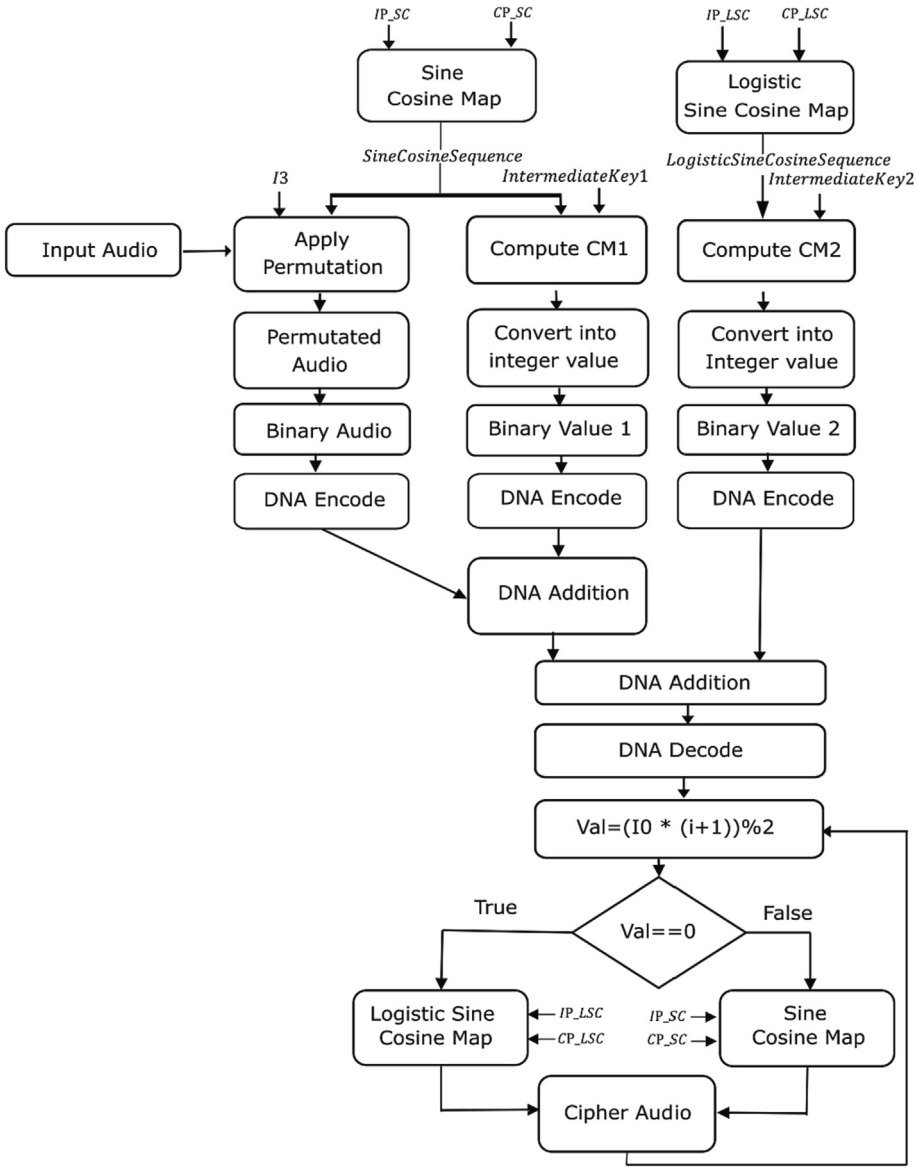


Fig. 3. General Architecture of Encryption Scheme.

equation (2) using the seed (IP_LSC) and control parameter (CP_LSC), stored to the *LogisticSineCosineSequence* variable.

$$IP_SC = 1/(hw(ShareKey) + 1) \tag{3}$$

$$CP_SC = 2.2 + ((I2 \oplus I3) \% 5) \tag{4}$$

$$IP_LSC = 1/(hw(InitalVector) + 1) \tag{5}$$

$$CP_LSC = 1/(hw(I3) + 1) \tag{6}$$

Also, two different pair of intermediate keys are generated using the *ShareKey* and $I0$, $I1$ vectors, as shown in Eqs. (7)–(11).

$$val = I0\%64 \tag{7}$$

$$IntermediateKey1 = ShareKey[val : val + 64] \tag{8}$$

$$SK2 = Rot(ShareKey, 2 * floor(hw(InitalVector)/2) + 1) \tag{9}$$

$$val1 = I1\%64 \tag{10}$$

$$IntermediateKey2 = SK2[val1 : val1 + 64] \tag{11}$$

5.2. Audio permutation

To perform the permutation process, the input audio is imported as 16-bit integer (int16) values. The full audio is stored as the *length* variable that is further divided into chunks of size 16-bit denoted by the variable *SequenceSize*. The permutation is applied on this *SequenceSize* variable using the chaotic sequence *SineCosineSequence*. Function 1 and Function 2 give pseudocode for the permutation operation. Also, Fig. 4 describes example of the permutation process. The stepwise description is for the same is given as:

Step 1: Initialize the variables *rem* and div that are produced using the following equations. The variable div is used for calculating the total number of the variables *SequenceSize*. The *rem* variable is used for calculating the remaining number, after splitting into chunks of 16.

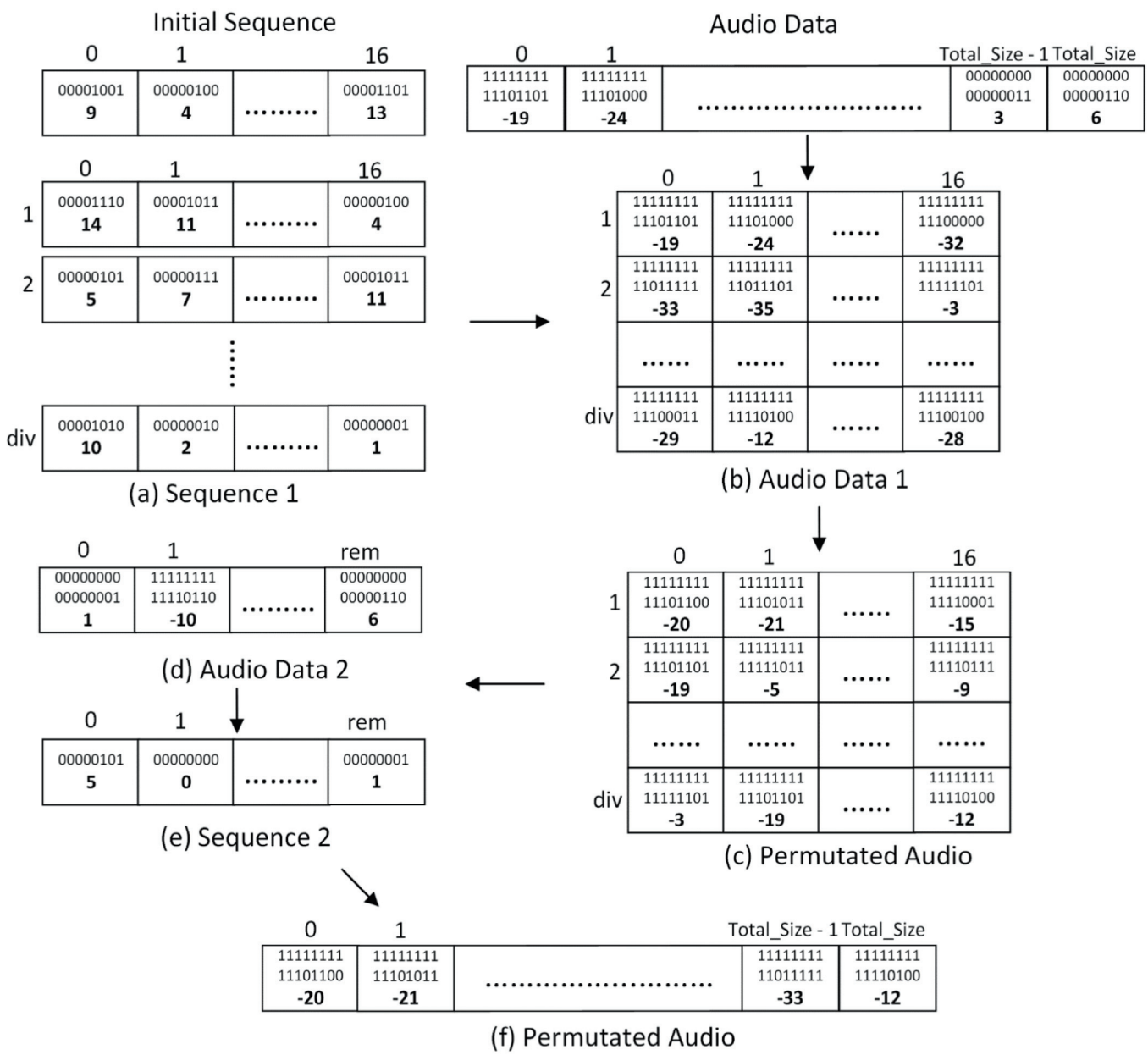


Fig. 4. Permutation Example.

$$rem = length\%16 \tag{12}$$

$$div = floor(length/16) \tag{13}$$

Step 2: The proposed encryption scheme generates matrices of 16 different sequence i.e., *Initial_Seq* for the permutation operation using the Function 1. This function takes the 32-bit Initial value ($I3$) as input, Sine-Cosine Map (*SineCosineSequence*), and total number of the sequence that are produced by the function (*SequenceSize*). It generates an output of the 16 different dynamic sequence for the permutation process (*InitialSequence*1). Then, the pixels of the audios are changed according the *InitialSequence*1, and this *InitialSequence*1 is modified with every iteration. Various steps for the permutation operation are:

(a) Initialize the two variables *arr* and *outputarr* of size equal to the size of the sequence (*SequenceSize*).

(b) For each sequence such that the value of *flag* is less than the *SequenceSize*, compute the *value_data* using the initial value ($I3$), SC Map (*SineCosineSequence*) and Increment variable (*inc*). The Increment variable are used for creating different values of the *value_data*.

$$value_data = int(I3 * inc * SineCosineSequence[inc]) \%SequenceSize \tag{14}$$

(c) Increase the value of *inc* variable by 1.

(d) Check whether the $arr[value_data]$ is previous generated or not using the equation (15). If the $arr[value_data]$ is not created, then save the *value_data* in the *outputarr* variable, modify the *flag* variable by 1 and update the $arr[value_data]$ as shown in algorithm.

$$arr[value_data] == False \tag{15}$$

Function 1: *Initial_Seq*

```
Input:IV3, SineCosineSequence, SequenceSize
CODE:
arr = np.zeros(SequenceSize, dtype = np.bool)
outputarr=np.zeros (SequenceSize,dtype='uint8')
inc = 1
flag = 0
WHILE(flag < SequenceSize) :
    value_data = int(IV3 * inc * SineCosineSequence[inc])
  %SequenceSize
    inc = inc + 1
    IFarr[value_data] == False :
      outputarr[flag − 1] = value_data
      flag = flag + 1
      arr[value_data] = True
    END IF
END WHILE LOOP
Output: outputarr
```

Function 2: Permutation

```
Input: Audio, IV3, SineCosineSequence
CODE:
rem = Length%16
div = int(Length/16)
InitialSequence1 = Initial_Seq(IV3, SineCosineSequence, 16)
InitialSequence2 = Initial_Seq(IV3, SineCosineSequence, rem)
audio_pos = 0
output = np.array([], dtype = np.int16)
Permutated_Val = np.zeros(16, dtype = np.int16)
For i lies between 0 to div do
  Sequence = np.roll(InitialSequence1, I3 * (i + 1))%16)
  Audio_16_val = Audio[audio_pos : audio_pos + 16]
  For i in range (16) do
    Permutated_Val[i] = Audio_16_val[Sequence[i]]
    output = np.concatenate((output, Permutated_Val))
    audio_pos = audio_pos + 16
  End Loop
End Loop
Audio_16_val = Audio[audio_pos : audio_pos + rem]
Permutated_Val1 = np.zeros(rem, dtype = np.int16)
For i in range(rem) do
  Permutated_Val1[i] = Audio_16_val[InitialSequence2[i]]
End Loop
output = np.concatenate((output, Permutated_Val1))
Output: output
```

Step 3: For each value of i from 0 to div, perform the following steps:

(a) Rotate the variable *InitialSequence*1 by $I3 * (i + 1)\%16$ and store in variable *Sequence*.

(b) Take 16 values of the audio data, and then permute these values according to the *Sequence* variable and store these value to the *output* variable.

Step 4: Permute the last remaining values of the audio using the second sequence (*InitialSequence*2), and store these values to the *output* variable.

5.3. Diffusion

To perform the diffusion process, Firstly, two random chaos sequences $CM1$ and $CM2$ are generated using pair of ($IntermediateKey1, SineCosineSequence$) and ($IntermediateKey2$, $LogisticSineCosineSequence$), respectively. The double DNA addition operation is applied on the DNA encoded binary values of $CM1$,

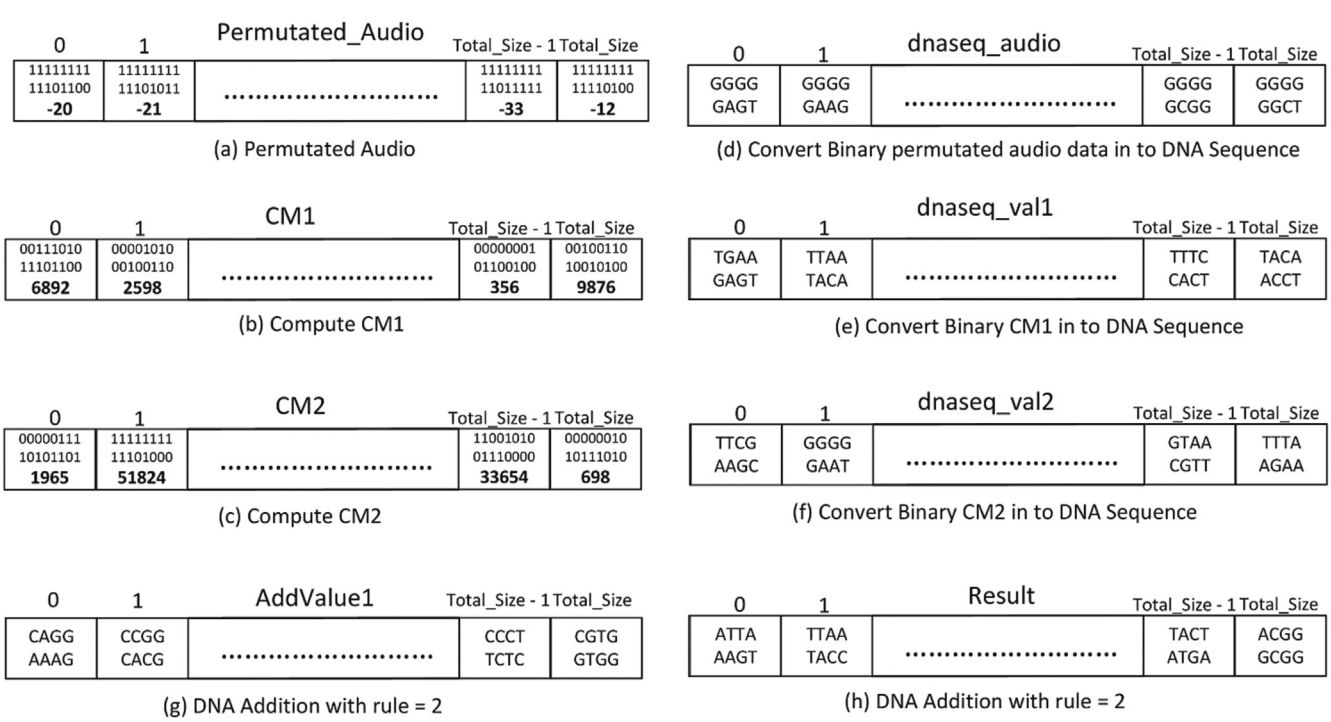


Fig. 5. Diffusion Example.

$CM2$, and permutated audio. As described by Fig. 3, in DNA encoding phase, the permuted binary image and binary valued sequences $CM1$, $CM2$ are converted into the DNA sequence using the operations specified in Table 1, and then double DNA addition operation is applied in these three sequences using the DNA addition operation specified in Table 2.

Fig. 5 shows the example of the DNA encoding and DNA addition used by the proposed audio encryption scheme. The binary form and DNA encoded form of permutated audio are shown by Fig. 5(a) and (d), respectively. For instance, binary form −20 represented as 1,111,111,111,101,100 is converted into DNA sequence GGGGGAGT. By using Table 1, binary value of −20 is split into two-two bits' combination i.e. (11 11 11 11 11 10 11 00). If rule is 2, then the 11 represent as G, 10 represents as A, 01 represents as C, and 00 represents as T, i.e., the binary value of −20 in DNA encoded form represents as GGGGGAGT.

Similarly, binary forms of $CM1$ and $CM2$ are shown in Fig. 5(b) and (c), respectively, and corresponding DNA sequences of these binary forms are shown by Fig. 5(e) and (f), respectively. The first DNA addition operation is applied between permutated audio DNA sequence, i.e., *dnaseq_audio* and DNA sequence of $CM1$, i.e., $dnaseq_val1$. The result of the same has been named as $AddValue1$ and is shown by Fig. 5(g). The double DNA operation's final result, named as variable *Result* is computed after applying the second DNA addition operation between the $dnaseq_val2$ of Fig. 5(f) and the $AddValue1$ of Fig. 5(e). The same has been shown by Fig. 5(h).

Finally, the dynamic diffusion, as explained in step 5 below, is performed by selecting one of the chaotic maps LSC map or SC map. This paper performs dynamic diffusion by choosing one of the two chaotic maps i.e., SC map & LSC map. The selection of the chaotic map is done using the "*val*" variables that are generated from the 32-bit Initial value and the position of the audio sample. Hence, the attacker gets it difficult to choose the correct chaotic map. Hence, this dynamic selection enhances the security of the encryption. The various steps followed for the diffusion process are as:

Step 1: Convert the audio data into 16-bit binary values, and initialise the rule according to the equation (16).

$$rule = Key\%8 \tag{16}$$

Step2: After that, the SC and LSC chaotic maps are converted into 16-bit integer values using the equation (17) - (18).

$$CM1 = (IntermediateKey1 * SineCosineSequence)/2^{16} \tag{17}$$

$$CM2 = (IntermediateKey2 * LogisticSineCosineMap)/2^{16} \tag{18}$$

Step 3: Convert the $CM1$ and $CM2$ into 16-bit binary values.
Step 4: Apply the double DNA addition operation as describes in Function 3.

(a) Convert the binary sequence of audio, SC, and LSC chaotic map into DNA Sequence using the equation (19) - (21).

$$dnaseq_audio = Binary_to_DNA_Seq(rule, binary_audio) \tag{19}$$

$$dnaseq_val1 = Binary_to_DNA_Seq(rule, binary_chaosValue1) \tag{20}$$

$$dnaseq_val2 = Binary_to_DNA_Seq(rule, binary_chaosValue2) \tag{21}$$

(b) Perform the DNA addition operation between the $dnaseq_audio$, $dnaseq_val1$, $dnaseq_val2$ as shown in equation (22)-(23).

$$AddValue1 = DNA_addition(dnaseq_audio, dnaseq_val1) \tag{22}$$

$$Result = DNA_addition(AddValue1, dnaseq_val2) \tag{23}$$

(c) Convert the DNA sequence into the binary value according to the equation (24).

$$dnaseq_output = DNA_2_binary(rule, Result) \tag{24}$$

Step 5: For each value of i from 0 to length, perform the following steps.

i. Select one chaotic map at ith position using the variable val as shown in equation (25). If the value of the variable val is 0, then select LSC map. Otherwise, select the SC chaotic map.

$$val = (I0 * (i + 1))\%2 \tag{25}$$

ii. Convert the *SineCosineSequence* and *LogisticSineCosine Sequence* into the integer value using the following equation.

$$I01 = int(I1 * SineCosineSequence) \tag{26}$$

$$I02 = int(I2 * LogisticSineCosineSequence) \tag{27}$$

iii. Generate the cipher audio using the SC Chaotic map and LSC map as shown in Eqs. (28)–(29), respectively. Function 4 shows the algorithm of the proposed audio encryption scheme.

$$CipherVoice[i] = (DNA_Voice_data[i] + I01[i] * (i + 1))\%Two_Power_16 \tag{28}$$

$$CipherVoice[i] = (DNA_Voice_data[i] + I02[i] * (i + 1))\%Two_Power_16 \tag{29}$$

Function 3:*Apply_dna*

Input: *Key*,
binary_audio, *binary_chaosValue*1, *binary_chaosValue*2
$rule = Key\%8$
$dnaseq_audio = Binary_to_DNA_Seq(rule, binary_audio)$
$dnaseq_val1 = Binary_to_DNA_Seq(rule, binary_chaosValue1)$
$dnaseq_val2 = Binary_to_DNA_Seq(rule, binary_chaosValue2)$
$AddValue1 = DNA_addition(dnaseq_audio, dnaseq_val1)$
$Result = DNA_addition(AddValue1, dnaseq_val2)$
$dnaseq_output = DNA_2_binary(rule, Result)$
Output: *dnaseq_output*

Function 4: Encryption

Input: *Audio, ShareKey*, InitialPar
CODE:
Initialization phase
$Length = len(Audio)$
$CipherVoice$ = $np.zeros(Length, dtype$ = 'int16')
$InitalVector_hex = hex(InitalVector)$
$I0 = int(InitalVector_hex[2:8+2], 16)$
$I1 = int(InitalVector_hex[8+2:16+2], 16)$
$I2 = int(InitalVector_hex[16+2:24+2], 16)$
$I3 = int(InitalVector_hex[24+2:32+2], 16)$
$val = I0\%64$
$IntermediateKey1 = ShareKey[val : val+32]$
$d = (2 * floor(hw(InitalVector)/2)) + 1$
$Tmp_Key = Left_Rotate(ShareKey, d)$
$val1 = I1\%64$
$IntermediateKey2 = Tmp_Key[val1 : val1+32]$
$IP_SC = 1/(hw(ShareKey)+1)$
$CP_SC = 2.2 + ((I2 \oplus I3)\ \%\ 5)$
$IP_LSC = 1/(hw(InitalVector)+1)$
$CP_LSC = 1/(hw(I3)+1)$
Chaotic Map Generation
$SineCosineSequence = SineCosineChaoticMap(IP_SC, CP_SC, Length)$
$LogisticSineCosineSequence = LogisticSineCosine(IP_LSC, CP_LSC, Length)$
Permutation phase
$Permutated_Audio = Permutation(Audio, I3, SineCosine\ Sequence)$
Diffusion phase
$CM1 = int((IntermediateKey1 * SineCosineSequence)/2^{16})$
$CM2 = int((IntermediateKey2 * LogisticSineCosineSequence)/2^{16})$
$binary_val1 = binary_repr(CM1, width = 16)$ *## Convert CM1 into binary values*
$binary_val2 = binary_repr(CM2, width = 16)$ *## Convert CM2 into binary values*
$binary_audio = binary_repr(Permutated_Audio, width = 16)$
$DNA_data = apply_dna(IntermediateKey32Bit2, binary_audio, binary_val1, binary_val2)$
Dynamic sequence Generation
$I01 = int(I1 * SineCosineSequence)$
$I02 = int(I2 * LogisticSineCosineSequence)$
$DNA_Voice_data = np.array(DNA_data, dtype = np.int16)$
For *i lies from* 0 *to Length* ***do***
  $val = (I0 * (i+1))\%2$
  IF $val == 0$:
    $CipherVoice[i] = (DNA_Voice_data[i] + I01[i] * (i+1))\%2^{16}$
  END IF
  ELSE :
    $CipherVoice[i] = (DNA_Voice_data[i] + I02[i] * (i+1))\%2^{16}$
  END ELSE
END FOR LOOP
Output: *CipherVoice*

5.4. Decryption

The decryption process is reverse to the encryption process. Fig. 6 describes the overall decryption process. The stepwise procedure for decryption process is as:

Step 1: Initialise the parameters for the decryption process.

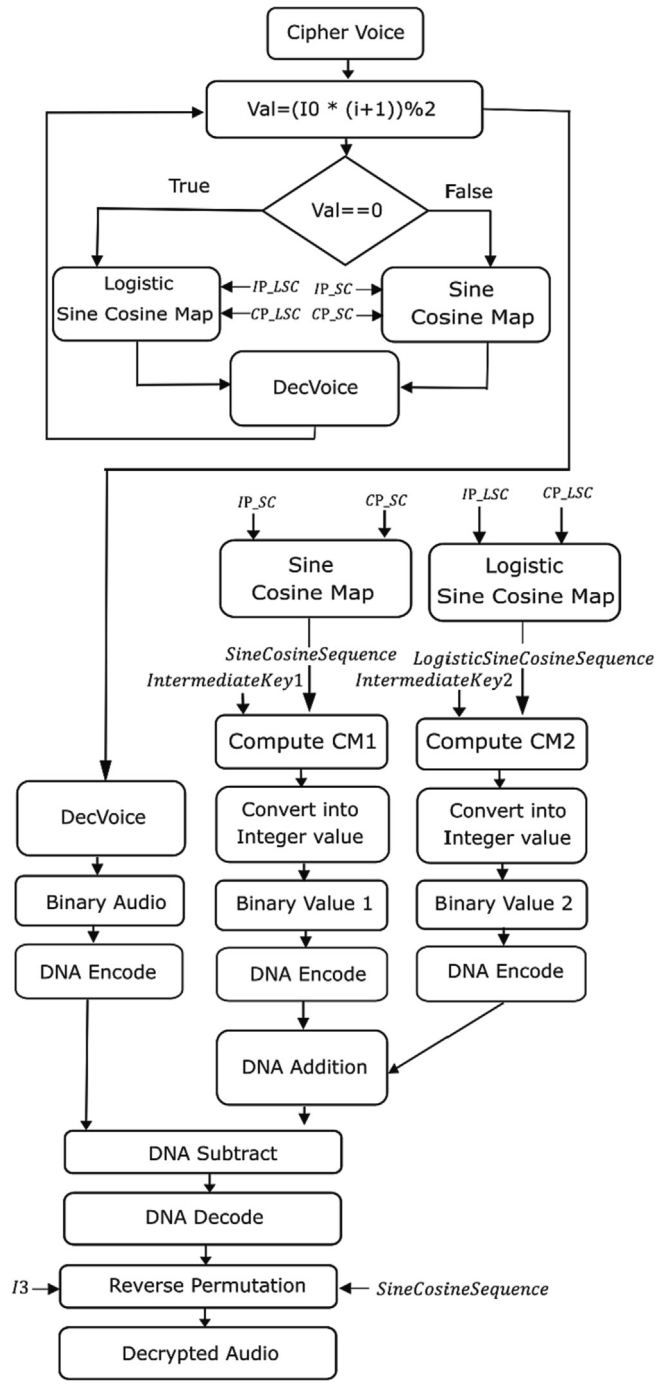


Fig. 6. Decryption Process.

Step 2: For each value of i from 0 to length, Select one chaotic map at i^{th} position using the variable val of Eq. (30). If the value of the variable val is 0, then compute the *DecVoice* using Eq. (31). Otherwise, computes the *DecVoice* using the Eq. (32).

$$val = (I0 * (i+1))\%2 \tag{30}$$

$$DecVoice[i] = (CipherVoice[i] - I01[i] * (i+1))\%Two_Power_16 \tag{31}$$

$$DecVoice[i] = (CipherVoice[i] - I02[i] * (i+1))\%Two_Power_16 \tag{32}$$

Table 3
Experimental audio.

Audio (.wav)	Size (Megabytes)	Length (Second)	Bitrate (Kilobytes per second)
Test1.wav [59]	4.42	26 Second	1411
Test2.wav [60]	7.03	41 Second	1411
Test3.wav [61]	6.39	38 Second	1411
Test4.wav [62]	8.07	48 Second	1411

Step 3: Perform the diffusion process using the DNA operation.
Step 4: Perform the reverse permutation and generate the decrypted image.

6. Experimental setup and results Analysis

This work has been performed using the Intel i7 processor with 512 GB SSD and 16 GB Ram. This experiment is carried out with python language using different packages like Scipy, Numpy, Matplotlib. Table 3 describes the detail of the audio files i.e., Test1, Test2, Test3, and Test4, that are used for evaluating the proposed audio encryption scheme. These audios are taken from the *mixkit.co* website. Test 1.wav [59], Test 2.wav [60], Test3.wav [61], and Test 4.wav [62] audios describe the sound of the keyboard typing, game show suspense waiting, Sea waves with birds' loop, and natural ambience with flowing water & birds, respectively.

6.1. Histogram Analysis

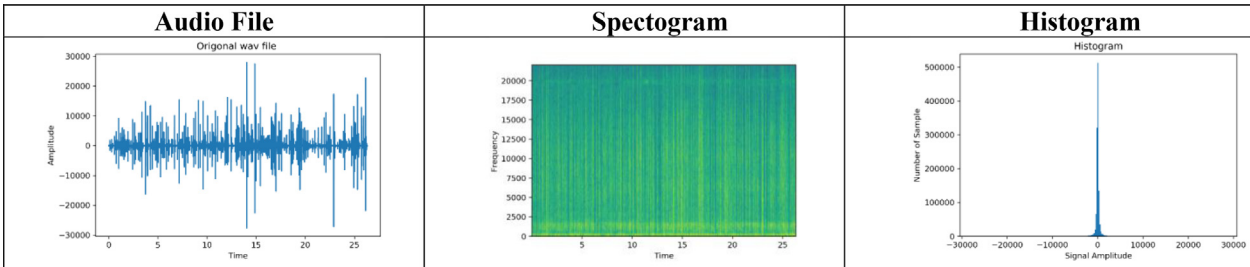
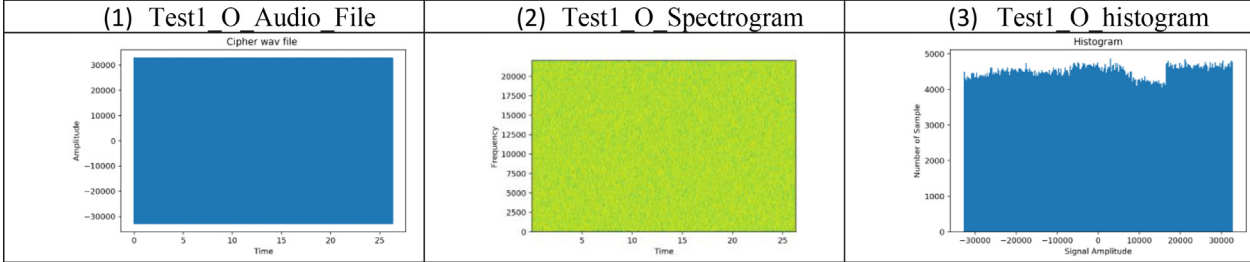
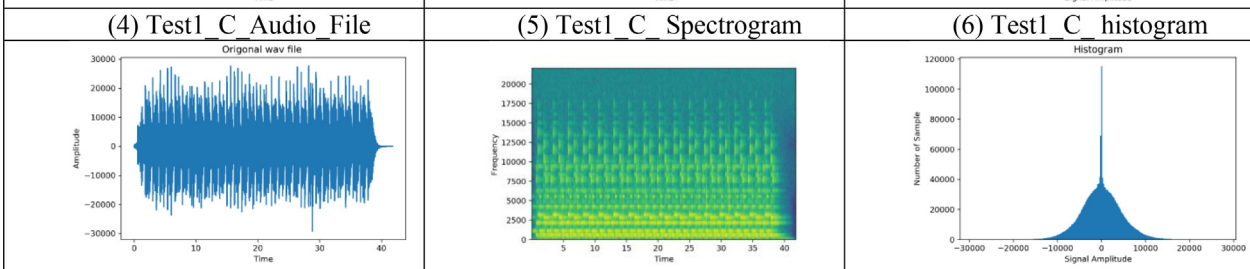
Histogram analysis shows the distribution of audio signal. This distribution is used by attackers to predict the intensity of audio signal. As a result, a uniformly distributed histogram is a necessity condition for an effective audio encryption strategy. Table 4 shows the analysis of the proposed scheme using the waveform of audio file, spectrogram, and histogram of input audios and cipher audios. The output result of waveform of audio file, spectrogram and histogram are represented using the filename_type_Audio_File, filename_type_Spectrogram, and Filename_type_histogram, respectively where "filename" represent as the name of input file i.e., Test1, Test2, Test3, Test4 and "type" represent as the original audio ("O") and cipher audio ("C"). It can be observed from the obtained results that the encrypted audio file has uniform structure for all three, waveform, histogram and spectrogram, representations. Hence, it is difficult for the attacker to identify the information.

6.1.1. Number of Sample Change Rate (NSCR)

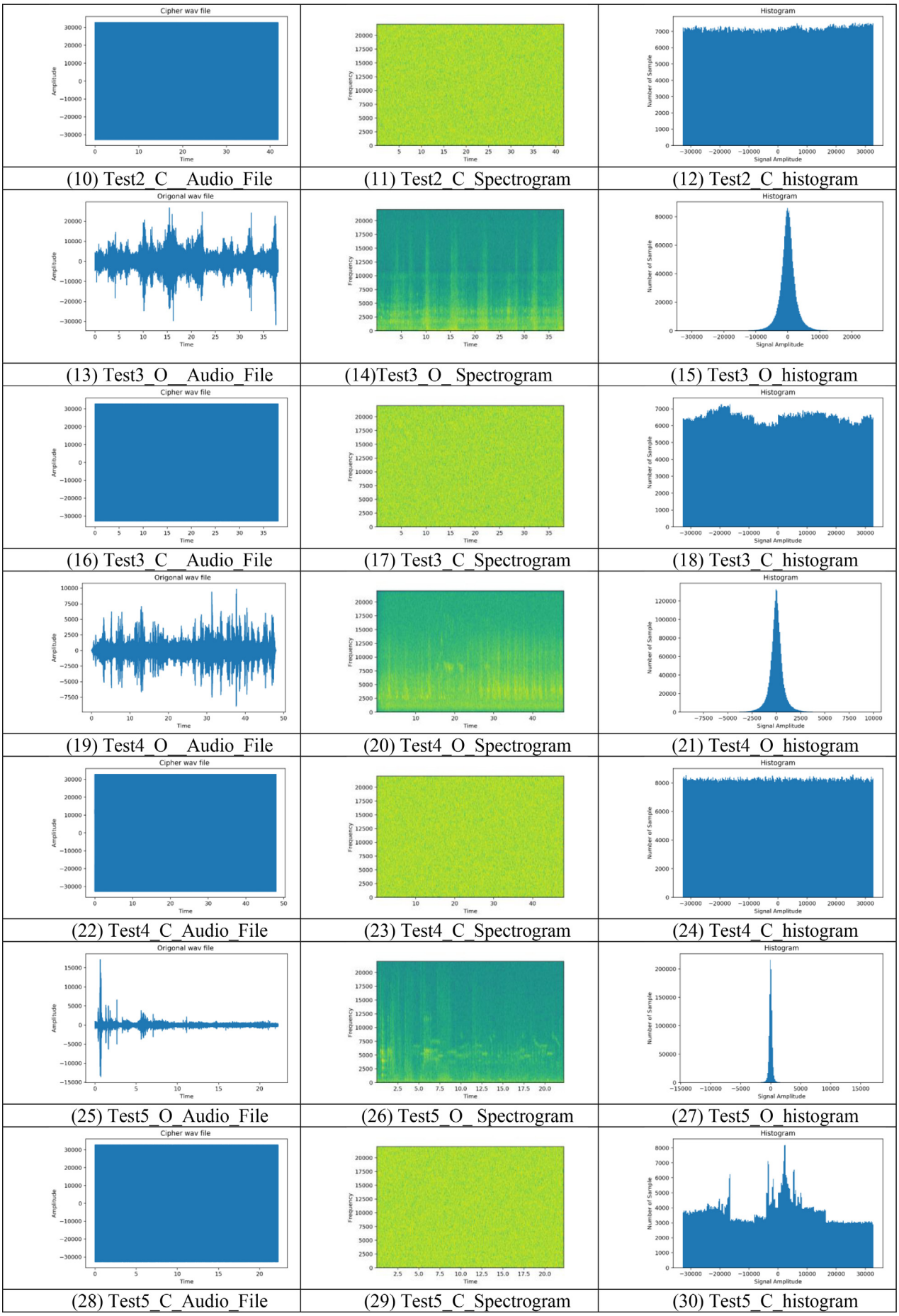
It determines the quality of encryption strategy of audio file. The NSCR calculates the difference of the sample values of the original and encrypted audio files. Equation (33) shows the mathematically formula of the NSCR.

$$NSCR = \frac{\sum_{i=1}^{N} D_i}{N} * 100 \tag{33}$$

Table 4
Analysis of proposed audio encryption.

Audio File	Spectogram	Histogram
		
(1) Test1_O_Audio_File	(2) Test1_O_Spectrogram	(3) Test1_O_histogram
		
(4) Test1_C_Audio_File	(5) Test1_C_ Spectrogram	(6) Test1_C_ histogram
		
(7) Test2_O_Audio_File	(8) Test2_O_ Spectrogram	(9) Test2_O_ histogram

(*continued on next page*)



(10) Test2_C__Audio_File (11) Test2_C_Spectrogram (12) Test2_C_histogram

(13) Test3_O__Audio_File (14)Test3_O_Spectrogram (15) Test3_O_histogram

(16) Test3_C__Audio_File (17) Test3_C_Spectrogram (18) Test3_C_histogram

(19) Test4_O__Audio_File (20) Test4_O_Spectrogram (21) Test4_O_histogram

(22) Test4_C_Audio_File (23) Test4_C_Spectrogram (24) Test4_C_histogram

(25) Test5_O_Audio_File (26) Test5_O_Spectrogram (27) Test5_O_histogram

(28) Test5_C_Audio_File (29) Test5_C_Spectrogram (30) Test5_C_histogram

$$D_i = \begin{cases} -1, x_i! = y_i \\ 0, otherwise \end{cases} \tag{34}$$

where x_i, y_i represent as the ith pixel of the input audio signal and cipher audio. Ideally, the value of the Number of Sample Change Rate is equal to 100. Table 5 shows the result of the Number of Sample Change Rate using the different types of the audios. The obtained result shows that the value of NSCR lies close to the ideal value.

6.1.2. Unified average Change Rate (UACI)

UACI displays the difference in average intensity between both the two audios. Encryption is recommended for the UACI's greater value. It is mathematically defined as:

$$UACI = \frac{1}{L}\left(\sum_{i,j} \frac{C_1(i,j) - C_2(i,j)}{2^{16} - 1}\right) * 100 \tag{35}$$

where C_1 and C_2 are defined as two different cipher audios. L represent as the total length of the audio sample. Table 5 shows the result of the UACI values using the different types of the audios. The obtained result shows that the values of the UACI lies close to 33.2.

6.1.3. Correlation Coefficient (CC)

The correlation coefficient (CC) measures the dependency between the two audio files [53,54]. This metrics measure the quality of the encryption scheme. It determines the relationship between the input audio and the encrypted audio file. Eq. (39) shows the mathematically formula of the CC.

$$r_{xy} = \frac{cov(x,y)}{\sqrt{T(x)}\sqrt{T(y)}} \tag{36}$$

$$T(x) = \frac{1}{Size}\sum_{pos=1}^{Size}(\mathrm{x_{pos}} - \mathrm{E(x)})^2 \tag{37}$$

$$E(x) = \frac{1}{Size}\sum_{pos=1}^{Size} x_{pos} \tag{38}$$

$$cov(x,y) = \frac{1}{Size}\sum_{pos=1}^{Size}(\mathrm{x_{pos}} - \mathrm{E(x)})(\mathrm{y_{pos}} - E(y)) \tag{39}$$

where *Size* is defined as the total size of the audio file. Ideally, the value of the CC lies close to 0. Table 5 presents the values of the CC of different audio files. The result shows that the CC of the original audio close to the 1. However, the CC of the cipher audios lies close to the zero.

6.1.4. Brute force attack

In this type of attack, the adversary performs an exhaustive search to calculate all the possible combinations for decrypting an encrypted audio. During the encryption/decryption process, the suggested scheme employs a 128-bit initial vector and a 128-bit shared key. As a result, the adversary needs 2^{128} combinations to estimate the initial values and the secret key, that is more than enough to withstand a brute force attack.

Table 5
Encryption parameter.

Audio	Correlation Coefficient (CC)		NSCR	UACI
	Original	Cipher		
Test1	0.8011	0.0053	99.9983	33.2
Test2	0.9710	−0.0026	99.9985	33.4
Test3	0.9652	−0.0194	99.9984	33.1
Test4	0.9153	0.0019	99.9980	33.15

6.1.5. Known/Chosen plaintext attacks

The adversary encrypts the numerous audios using the same share keystream in a chosen-plaintext attack. The adversary's goal is to obtain a shared key that are used in the encryption procedure. The suggested encryption approach generates the keystream using a combination of key and seed value. The keystream is reliant on the shared key and random vector throughout the encryption procedure. As a result, the other cipher audio does not use the same keystream to decode. Hence, this method prevents the chosen attack.

6.1.6. Peak signal to noise Ratio

PSNR is described as the fraction of the maximum power capacity to the power of the encrypt speech. It is used to test the quality of the signal. Eq. (40) shows the mathematical formula of the peak signal to noise ratio.

$$PSNR = 10\log\frac{MAX^2}{MSE} \tag{40}$$

where x, k is defined as the input signal, encrypted signal respectively. The difference of energy of original and cipher signal represents as $x - k$. The Mean Square Error (MSE) is described as the amount of bit error present in the audio encryption, when some amount of noise added to the encryption. The MSE is calculated using the difference of two audio signal i.e., input audio and the noisy decrypted audio signal. Mathematically, it is defined as below:

$$MSE = \frac{\sum_{i,j}(A[i,j] - B[i,j])^2}{Size} \tag{41}$$

Here, *Size* refers as the total number of audio pixels. A and B refers as the input audio signal and the noisy decrypted audio signal, respectively. The BCR values obtained using the suggested technique are shown in Table 6. The obtained result shows that the after adding different level of "salt and pepper" noise (1 %, 2 %, 5 % and 10 %) in the cipher audio, the nearly 2 %, 4 %, 10 % and 20 % of pixel modified for all the four audios.

6.1.7. Bit corrected Ratio (BCR)

The BCR is determined by calculating the number of bit errors in both the original and decrypted audio. BCR is expressed mathematically as follows:

$$BCR = 1 - \frac{\sum_{i}^{Length} A(i) \oplus D(i)}{size} \tag{42}$$

where $A(i)$ and $D(i)$ represent the value of data at position i of original audio, and received audio to the receiver after adding the Gaussian noise. The Bit Correct Ratio has a range of [0, 1]. The optimal BCR value is 1. Table 6 displays the BCR values achieved using the proposed technique. The results show that adding 1 % "salt and pepper" noise to the encryption audio impacted around 2 % of the pixels in all four audios.

6.1.8. Local Shannon Entropy

The local Shannon Entropy is calculated using the average of the individual "k" component of the Shannon entropy [55]. Shannon Entropy determines the amount of the uncertainty in a random variable. For a 16-bit Audio, the ideal value of the Shannon Entropy is equal to 16. Figs. 7 and 8 show the value of the individual component of the local Shannon Entropy for input audio and cipher audio, respectively.

Table 6
Analysis of decrypted audio.

Input Audio	Parameter	Salt and Pepper Noise			
		1 %	2 %	5 %	10 %
Test1	BCR	0.9800	0.9601	0.9000	0.8003
	PSNR	27.7384	24.7767	20.7849	17.7931
Test2	BCR	0.9799	0.9599	0.8996	0.8004
	PSNR	27.7497	24.7331	20.7746	17.7854
Test3	BCR	0.9800	0.9595	0.8999	0.8000
	PSNR	27.7934	24.7250	20.7984	17.7885
Test4	BCR	0.9799	0.9598	0.8999	0.7998
	PSNR	27.5043	24.5005	20.5332	17.5184

6.1.8.1. National institute of standards and technology (NIST) Test. The efficiency of the proposed encryption approach is examined using the fifteen different NIST tests [56]. The p-values of the various NIST tests are shown in Table 7, which demonstrates that the proposed audio cryptosystem passes every test.

6.1.8.2. Evaluation of speed of proposed scheme. Table 8 shows the computation analysis of the proposed algorithm using the Test1, Test2, Test3, and Test4 audios. The total computation time for the Test1, Test2, Test3 and Test4 audios are 151.2 s, 276.4 s, 233.6ss7 second, and 297.32 s, respectively. Hence, the speed of the algorithm for computing the number of bytes in 1 s using the four different audios (Test1, Test2, Test3, Test4) are 29.93, 26.04, 28.00 and 27.79, respectively.

6.2. Comparative Analysis with existing methods

This section compares the proposed method to current audio encryption algorithms in terms of several parameters. Table 9 compares the values obtained in published works of current schemes to the values obtained using the proposed technique for the different parameters, including key space, NSCR, CC, PSNR, and BCR. The comparison suggests that the proposed technique is secure, effective, and capable of withstanding a range of threats.

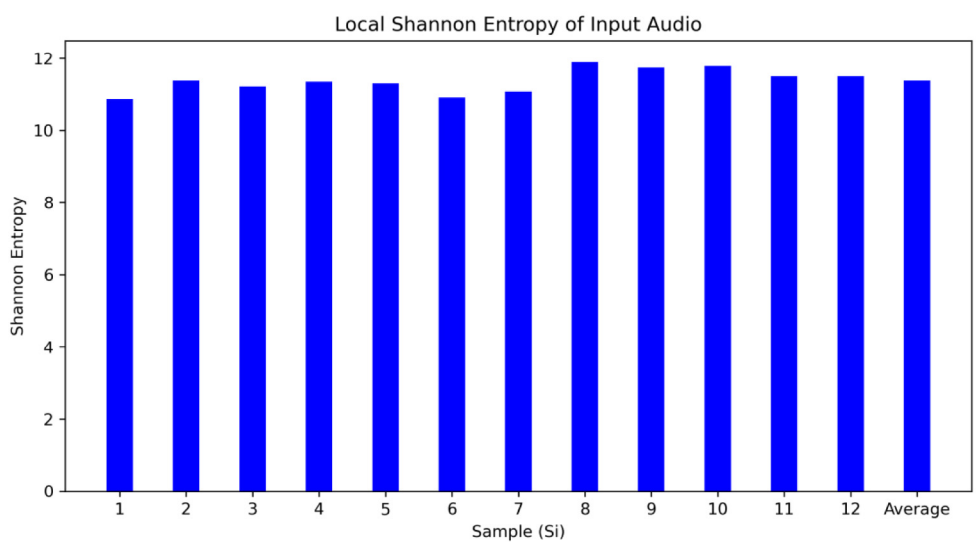


Fig. 7. Local Shannon Entropy of Input Audio.

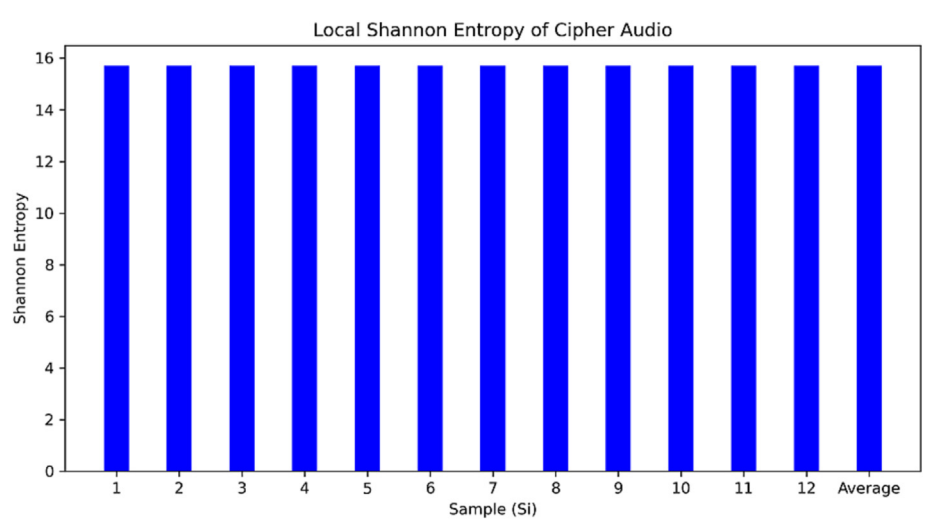


Fig. 8. Local Shannon Entropy of Cipher Audio.

Table 7
NIST test.

NIST Test	P-value		Pass (✓)/Fail (X)
Approximate Entropy	0.07079		✓
Binary Matrix Rank	0.21218		✓
Cumulative Sums Test	0.74764		✓
Dft Test	0.13760		✓
Frequency within block	0.43641		✓
Linear Complexity Test	0.79703		✓
Longest Run ones in a Block	0.22519		✓
Maurers Universal set	0.60782		✓
Monobit Test	0.61963		✓
Non overlapping Template Matching test	0.99985		✓
Overlapping Template Matching Test	0.54373		✓
Random Excursion Test	−4	0.733381	✓
	−3	0.914555	✓
	−2	0.708913	✓
	−1	0.650816	✓
	1	0.801584	✓
	2	0.588867	✓
	3	0.628372	✓
	4	0.395299	✓
Random Excursion Variant Test	−9	0.105469	✓
	−8	0.118212	✓
	−7	0.214155	✓
	−6	0.235584	✓
	−5	0.138226	✓
	−4	0.187949	✓
	−3	0.543441	✓
	−2	0.698271	✓
	−1	0.837274	✓
	1	0.591163	✓
	2	0.167009	✓
	3	0.144568	✓
	4	0.266714	✓
	5	0.319586	✓
	6	0.423561	✓
	7	0.535248	✓
	8	0.674373	✓
	9	0.686032	✓
Run Test	0.67736	✓	
Serial Test	0.0708	✓	

Table 8
Computation Time Analysis.

Audio File (.wav)	File Size (Megabytes)	Encryption Time (second)	Speed (Kilobyte/second)
Test1	4.42	151.20	29.93
Test2	7.03	276.40	26.04
Test3	6.39	233.67	28.00
Test4	8.07	297.32	27.79

Table 9
Comparatively Analysis With Existing Approach.

Work	Key Space	Speed (s/ Kilobits)	Parameters						
			NSCR (Encryption)	UACI (Encryption)	CC (Original)	CC (Encrypted)	PSNR (Encrypted)	PSNR (Decrypted)	BCR (Decrypted)
Lima et al. [57]	256 bit	–	99.9992	33.2924	0.6405	0.0021	–	–	–
Kate et al. [2]	256 bit	0.22	–	–	–	−0.0009	−7.80	–	–
Wang et al. [44]	–	0.034	**100**	33.3782	1.0000	−0.0034	−8.5007	–	–
Sasikaladevi et al. [58]	–	–	98.000	33.000	0.997	0.0133	–	1.6060e + 03	–
Proposed Approach	128-bit	**0.0041**	**99.9993**	**33.45**	0.9710	0.0019	−10.8877	30.8082*	**0.9900***

* 0.5% of Salt and Pepper Noise.

7. Conclusion & future work

The proposed work describes a novel encryption strategy that uses the LSC map, SC Map, and double DNA operation. This strategy uses the 128-bit initial vector and secret key to generate the control parameter and initial seed values for the chaotic maps. The permutation is carried out after applying the SC map and share key during the first phase of encryption process. The permuted audio is dynamically diffused using the DNA addition operation, and choice of one of the said chaotic maps in the second phase of the encryption method. The encryption strategy has been examined using the various existing metrics like MSE, PSNR, Histogram Analysis, CC, NSCR, speed and time analysis. The work can be further extended for the other multimedia applications such as images and videos. Also, improving time efficiency of the proposed audio cryptosystem is an important aspect that can be considered as future extension of the proposed work.

Data availability

I have given the link of data in references

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Naskar PK, Paul S, Nandy D, Chaudhuri A. DNA encoding and channel shuffling for secured encryption of audio data. Multimed Tools Appl 2019;78(17):25019–42.
[2] H. Kakaei Kate, J. Razmara, A. Isazadeh, Correction to: A Novel Fast and Secure Approach for Voice Encryption Based on DNA Computing (3D Research, (2018), 9, 2, (17), 10.1007/s13319-018-0167-x), 3D Res. 9 (2018) 1–11. 10.1007/s13319-018-0176-9.
[3] I. El Hanouti, H. El Fadili, Correction to: Security analysis of an audio data encryption scheme based on key chaining and DNA encoding (Multimedia Tools and Applications, (2021), 80, 8, (12077-12099), 10.1007/s11042-020-10153-8), Multimed. Tools Appl. 80 (2021) 12101. 10.1007/s11042-021-10592-x.
[4] Liu H, Kadir A, Li Y. Audio encryption scheme by confusion and diffusion based on multi-scroll chaotic system and one-time keys. Optik (Stuttg) 2016;127:7431–8. https://doi.org/10.1016/j.ijleo.2016.05.073.
[5] Kordov K. A novel audio encryption algorithm with permutation-substitution architecture. Electronics 2019;8:530.
[6] George SN, Augustine N, Pattathil DP. Audio security through compressive sampling and cellular automata. Multimed Tools Appl 2015;74:10393–417. https://doi.org/10.1007/s11042-014-2172-2.
[7] Bisht A, Dua M, Dua S, Jaroli P. A color image encryption technique based on bit-level permutation and alternate logistic maps. J Intell Syst 2020;29:1246–60. https://doi.org/10.1515/jisys-2018-0365.
[8] K. Suneja, S. Dua, M. Dua, A review of chaos based image encryption, in: Proc. 3rd Int. Conf. Comput. Methodol. Commun. ICCMC 2019, IEEE, 2019: pp. 693–698. 10.1109/ICCMC.2019.8819860.
[9] Hua Z, Zhou Y, Pun CM, Chen CLP. 2D Sine Logistic modulation map for image encryption. Inf Sci (Ny) 2015;297:80–94. https://doi.org/10.1016/j.ins.2014.11.018.
[10] Hua Z, Zhou Y. Image encryption using 2D Logistic-adjusted-Sine map. Inf Sci (Ny) 2016;339:237–53. https://doi.org/10.1016/j.ins.2016.01.017.
[11] Hui Y, Liu H, Fang P. A DNA image encryption based on a new hyperchaotic system, Multimed. Tools Appl 2021:1–25.
[12] Guesmi R, Farah MAB. A new efficient medical image cipher based on hybrid chaotic map and DNA code. Multimed Tools Appl 2021;80:1925–44. https://doi.org/10.1007/s11042-020-09672-1.
[13] K. Kumar, S. Roy, U. Rawat, I. Mishra, An Efficient Image Encryption Technique Using Logistic Map and 2D-TCLM, Commun. Comput. Inf. Sci. 1599 CCIS (2022) 87–96. 10.1007/978-3-031-15784-4_7.
[14] Dua M, Kumar A, Garg A, Garg V. Multiple image encryption approach using non linear chaotic map and cosine transformation. Int J Inf Technol 2022;14:1627–41. https://doi.org/10.1007/s41870-022-00885-1.
[15] Wu Y, Noonan JP, Yang G, Jin H. Image encryption using the two-dimensional logistic chaotic map. J Electron Imaging 2012;21:13014.
[16] Sharma M. Image encryption based on a new 2D logistic adjusted logistic map. Multimed Tools Appl 2020;79(1-2):355–74.
[17] Yu F, Qian S, Chen X, Huang Y, Cai S, Jin J, et al. Chaos-based engineering applications with a 6d memristive multistable hyperchaotic system and a 2D SF-SIMM hyperchaotic map. Complexity 2021;2021. https://doi.org/10.1155/2021/6683284.
[18] Wang X, Xu D. Image encryption using genetic operators and intertwining logistic map. Nonlinear Dyn 2014;78:2975–84. https://doi.org/10.1007/s11071-014-1639-z.
[19] Dua M, Suthar A, Garg A, Garg V. An ILM-cosine transform-based improved approach to image encryption. Complex Intell Syst 2021;7(1):327–43.
[20] Dua M, Wesanekar A, Gupta V, Bhola M, Dua S. Differential evolution optimization of intertwining logistic map-DNA based image encryption technique. J Ambient Intell Humaniz Comput 2020;11(9):3771–86.
[21] Van Gorder RA. Shil'nikov chaos in the 4D Lorenz-Stenflo system modeling the time evolution of nonlinear acoustic-gravity waves in a rotating atmosphere. Nonlinear Dyn 2013;72:837–51. https://doi.org/10.1007/s11071-013-0756-4.
[22] Zhou P, Yang F. Hyperchaos, chaos, and horseshoe in a 4D nonlinear system with an infinite number of equilibrium points. Nonlinear Dyn 2014;76:473–80. https://doi.org/10.1007/s11071-013-1140-0.
[23] Zhang F, Yang G, Zhang Y, Liao X, Zhang G. Qualitative study of a 4D chaos financial system. Complexity 2018;2018. https://doi.org/10.1155/2018/3789873.
[24] Zhang F, Chen R, Wang X, Chen X, Mu C, Liao X. Dynamics of a new 5D hyperchaotic system of lorenz type. Int J Bifurc Chaos 2018;28:1850036. https://doi.org/10.1142/S0218127418500360.
[25] khan N, Muthukumar P. Transient chaos, synchronization and digital image enhancement technique based on a novel 5d fractional-order hyperchaotic memristive system, circuits. Syst Signal Process 2022;41(4):2266–89.
[26] Yu F, Liu L, Qian S, Li L, Huang Y, Shi C, et al. Chaos-based application of a novel multistable 5d memristive hyperchaotic system with coexisting multiple attractors. Complexity 2020;2020. https://doi.org/10.1155/2020/8034196.
[27] Mezatio BA, Motchongom MT, Wafo Tekam BR, Kengne R, Tchitnga R, Fomethe A. A novel memristive 6D hyperchaotic autonomous system with hidden extreme multistability. Chaos Solitons Fractals 2019;120:100–15. https://doi.org/10.1016/j.chaos.2019.01.015.
[28] Al-Azzawi SF, Al-Obeidi AS. Chaos synchronization in a new 6D hyperchaotic system with self-excited attractors and seventeen terms. Asian-European J Math 2021;14:2150085. https://doi.org/10.1142/S1793557121500856.
[29] Pareek NK, Patidar V, Sud KK. Image encryption using chaotic logistic map. Image Vis Comput 2006;24(9):926–34.
[30] Belazi A, El-Latif AAA. A simple yet efficient S-box method based on chaotic sine map. Optik (Stuttg) 2017;130:1438–44. https://doi.org/10.1016/j.ijleo.2016.11.152.
[31] Li C, Luo G, Qin K, Li C. An image encryption scheme based on chaotic tent map. Nonlinear Dyn 2017;87:127–33. https://doi.org/10.1007/s11071-016-3030-8.
[32] Kumar A, Dua M. Novel pseudo random key & cosine transformed chaotic maps based satellite image encryption. Multimed Tools Appl 2021;80:27785–805. https://doi.org/10.1007/s11042-021-10970-5.
[33] Kumar A, Dua M. A GRU and chaos-based novel image encryption approach for transport images. Multimed Tools Appl 2022. https://doi.org/10.1007/s11042-022-13902-z.
[34] Banu A, Amirtharajan SR. A robust medical image encryption in dual domain: chaos-DNA-IWT combined approach. Med Biol Eng Comput 2020;58:1445–58. https://doi.org/10.1007/s11517-020-02178-w.
[35] Yu W, Liu Ye, Gong L, Tian M, Tu L. Double-image encryption based on spatiotemporal chaos and DNA operations. Multimed Tools Appl 2019;78(14):20037–64.
[36] Telem ANK, Fotsin HB, Kengne J. Image encryption algorithm based on dynamic DNA coding operations and 3D chaotic systems. Multimed Tools Appl 2021:1–31.
[37] Dua M, Wesanekar A, Gupta V, Bhola M, Dua S. Differential evolution optimization of intertwining logistic map-DNA based image encryption technique. J Ambient Intell Humaniz Comput 2020;11:3771–86. https://doi.org/10.1007/s12652-019-01580-z.
[38] Pankaj S, Dua M. A novel ToCC map and two-level scrambling-based medical image encryption technique. Netw Model Anal Heal Informatics Bioinforma 2021;10:48. https://doi.org/10.1007/s13721-021-00324-4.
[39] Yang YG, Tian J, Sun SJ, Xu P. Quantum-assisted encryption for digital audio signals. Optik (Stuttg) 2015;126:3221–6. https://doi.org/10.1016/j.ijleo.2015.07.082.
[40] Sheu LJ. A speech encryption using fractional chaotic systems. Nonlinear Dyn 2011;65:103–8. https://doi.org/10.1007/s11071-010-9877-1.
[41] Albahrani EA, Alshekly TK, Lafta SH. A review on audio encryption algorithms using chaos maps-based techniques. J Cyber Secur Mobil 2022;11:53–82. https://doi.org/10.13052/jcsm2245-1439.1113.
[42] Kaur G, Singh K, Gill HS. Chaos-based joint speech encryption scheme using SHA-1. Multimed Tools Appl 2021;80:10927–47. https://doi.org/10.1007/s11042-020-10223-x.
[43] Stoyanov B, Ivanova T. Novel implementation of audio encryption using pseudorandom byte generator. Appl Sci 2021;11:10190. https://doi.org/10.3390/app112110190.
[44] Wang X, Su Y. An audio encryption algorithm based on DNA coding and chaotic system. IEEE Access 2020;8:9260–70. https://doi.org/10.1109/ACCESS.2019.2963329.
[45] Li H, Wang Y, Zuo Z. Chaos-based image encryption algorithm with orbit perturbation and dynamic state variable selection mechanisms. Opt Lasers Eng 2019;115:197–207. https://doi.org/10.1016/j.optlaseng.2018.12.002.

[46] Sathiyamurthi P, Ramakrishnan S. Speech encryption using chaotic shift keying for secured speech communication, Eurasip. J Audio, Speech, Music Process 2017;2017:1–11. https://doi.org/10.1186/s13636-017-0118-0.
[47] Ghasemzadeh A, Esmaeili E. A novel method in audio message encryption based on a mixture of chaos function. Int J Speech Technol 2017;20:829–37. https://doi.org/10.1007/s10772-017-9452-y.
[48] Roy A, Misra AP. Audio signal encryption using chaotic Hénon map and lifting wavelet transforms. Eur Phys J Plus 2017;132:1–10. https://doi.org/10.1140/epjp/i2017-11808-x.
[49] Kalpana M, Ratnavelu K, Balasubramaniam P. An audio encryption based on synchronization of robust BAM FCNNs with time delays. Multimed Tools Appl 2019;78(5):5969–88.
[50] Dua M, Wesanekar A, Gupta V, Bhola M, Dua S. Color image encryption using synchronous CML-DNA and weighted bi-objective genetic algorithm, in. ACM Int Conf Proceeding Ser 2019:121–5. https://doi.org/10.1145/3361758.3361780.
[51] Hua Z, Zhou Y, Huang H. Cosine-transform-based chaotic system for image encryption. Inf Sci (Ny) 2019;480:403–19. https://doi.org/10.1016/j.ins.2018.12.048.
[53] Cao C, Sun K, Liu W. A novel bit-level image encryption algorithm based on 2D-LICM hyperchaotic map. Signal Process 2018;143:122–33. https://doi.org/10.1016/j.sigpro.2017.08.020.
[54] Cao W, Mao Y, Zhou Y. Designing a 2D infinite collapse map for image encryption. Signal Process 2020;171:. https://doi.org/10.1016/j.sigpro.2020.107457107457.
[55] Wu Y, Zhou Y, Saveriades G, Agaian S, Noonan JP, Natarajan P. Local Shannon entropy measure with statistical tests for image randomness. Inf Sci (Ny) 2013;222:323–42. https://doi.org/10.1016/j.ins.2012.07.049.
[56] Zaman J, Ghosh R. Review on fifteen Statistical Tests proposed by NIST. Ijtpc 2012;1:18–31. , http://www.doaj.org/doaj?func=fulltext&aId=1419927.
[57] Lima JB, Da Silva Neto EF. Audio encryption based on the cosine number transform, Multimed. Tools Appl 2016;75:8403–18. https://doi.org/10.1007/s11042-015-2755-6.
[58] Sasikaladevi N, Geetha K, Venkata Srinivas KN. A multi-tier security system (SAIL) for protecting audio signals from malicious exploits. Int J Speech Technol 2018;21:319–32. https://doi.org/10.1007/s10772-018-9510-0.
[59] Keyboard typing, "https://mixkit.co/free-sound-effects/keyboard/", accessed date=28-3-2022.
[60] Game show Suspense Waiting, "https://mixkit.co/free-sound-effects/suspense-music/", accessed date=28-3-2022.
[61] Sea waves with birds loop, "https://mixkit.co/free-sound-effects/sea/", accessed date=28-3-2022.
[62] Natural ambience with flowing water and birds, "https://mixkit.co/free-sound-effects/water/", accessed date=28-3-2022.