International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 10, No.1 (Jan-2021)

http://dx.doi.org/10.12785/ijcds/100117

A (k, n) Audio Secret Sharing with Share Authentication

Sachin Kumar Singh[1] and Mainejar Yadav [2]

[1]*Computer Science and Engineering Department, Rajkiya Engineering College Sonbhadra, Uttar Pradesh, India.*

Received 30 Apr. 2020, Revised 8 Aug. 2020, Accepted 29 Sep. 2020, Published 1 Jan. 2021

Abstract: In our day to day conversation audio is the most practiced way of communication. Either it can be face to face or digital like over phone or voice messages. Many part of our digital conversation may consist of important/sensitive information. Hence the security and confidentiality of this communication must be assured. The secret is usually transmitted in the form of a cipher, which is encrypted with a key, and the key is required for revealing the secret. In this case, the key becomes a single point of failure. For enhancing the security, the secret is sent in multiple units called shares. Here, an approach is proposed for the same. The secret is divided into *n* units, and a minimum threshold *k* is set. So secret can only be revealed when *k* shares are available. The authenticity of the shares can also be verified to detect if the shares have been tampered or not. This scheme can be used for communication channels and secure transmission of audio. No cover is required to transmit the share. If in case some of the shares are lost or get corrupted we may be able to reconstruct the secret. This reconstruction is possible even without the lost shares if shares available even after the loss are greater than or equal to *k.*

Keywords: Secret Sharing, Audio Security, Audio Communication, Correlation Coefficient, Integrity

1. INTRODUCTION

In the modern age, the security of information has become a significant issue. It is conceivable to copy computerized digital data a million-overlay and, through the internet, disseminate it over the whole world in seconds. Data digitization has expanded the exchange of personal information over the internet. Information storage and transmission attract a high number of intruders. While transmitting and storing the data, it is crucial to prevent data from vandalizing, and the information should not be lost. Information protection is therefore required to protect the confidentiality, integrity, and accessibility of the data.

When we talk about communication audio is the most practiced way, it is used by people in their day to day life and also used in every sector for information distribution. People make audio conversations over the phone, video call, voice messages etc. which may contain sensitive information and can be recorded and stored as well. Even sensitive sector like defense also extensively uses audio for relaying information. All these conversations can be wiretapped, which raises security and privacy issues. It is very necessary to secure these conservations, making it necessary to devise secure transmission methods.

There are various conventional and complex cryptography methods for securing these data via encryption. Encryption offers security but there are some problems associated with it. Usually the time and computation required for encrypting and decrypting the data is quite significant and if the encryption key gets lost or stolen by the attacker, these schemes can become a single point failure. Hence, a secret sharing scheme was implemented to ensure the security of such data.

This scheme of Secret Sharing was introduced by Shamir *et al.* [3] and Blakely *et al.* in 1979 [1]. In *(n, n)* secret sharing scheme secret is divided into *n* shares and *n* are required for the retrieval of secret information [19-21] and in *(k, n)* where *k* is the threshold value and out of *n* shares at least *k* shares are required for the retrieval of secret information.

There are various approaches to secure the audio data as proposed by researchers, but still, a completely effective approach has not been introduced yet. Here we are proposing a Verifiable *(k, n)* Audio Secret Sharing (VASS) scheme, which generates *n* share possessing *k* threshold using circular shift operation. The *(k, n)* VASS scheme proposed here is suitable as a secure communication and transmission method. This scheme does not require any cover audio for the transmission of the shares. Also, at the receivers end, the integrity verification can be performed for each share.

E-mail: sachin.ls@outlook.com , rahulit1210@gmail.com

The structure of the remaining paper is as follows:

A literature survey is given in section 2. Section 3 describes the proposed approach. The analysis of the experimental results is provided in section 4. Conclusion of the proposed work discussed in section 5.

2. LITERATURE REVIEW

Audio secret sharing (ASS) is a more secure way for the transmission or storage of audio data. A brief review of the approaches for the ASS scheme is presented in this section.

Sound/audio can be considered as a wave of energy that passes through a particular medium (here we are only considering air as a medium). Sound wave consist of high pressure condensations and low pressure rarefactions. Wave interference happens while moving when two wave touch. In a particular medium when two waves occur simultaneously either there high pressure condensations and low pressure rarefactions coincide with other wave high pressure condensations and low pressure rarefactions respectively, the wave amplifies and the sound become louder or the high pressure condensations and low pressure rarefactions coincide with low pressure rarefactions and high pressure condensations respectively, the wave diminishes, results low sound. The scenario in which the sound intensify and become louder is called constructive interference and in which it diminishes or cancel each other is called destructive interference. The early techniques of ASS were based on this property.

A (2, 2) audio secret sharing scheme with objective to hide a binary message through high-quality audio was proposed by Y. Desmedt *et al.* [13]. The binary secret was embedded in the high-quality audio. The binary message is a secret and the high-quality audio in which it is embedded act as cover. During the share generation, the secret is embedded and two high-quality stego-audio shares are generated. For reviling the secret these two shares are played together. This scheme exploits the interference property of wave. The first share is generated by randomly choosing a phase and generation of second share is based on the bit need to be embedded and it uses exclusive-or. This offers perfect secrecy as it is based on one time pad [5, 12]. For embedding/encoding computation is required, for reviling/decoding the secret no computation is required. This scheme was only limited to binary secret and audio secret can`t be embedded and the total number of covers required was $\lceil \log_2 n \rceil$, these comes out to be the major disadvantage of this scheme. The efficiency offered by this scheme was also poor as the total size of all the shares was too large as compare to the size of the secret.

Prabir *et al.* [8] proposed a method in which ANDing and ORing are used for generation of share and regeneration of audio secret. Each and every share which is generated will only have a part of original data. During the algorithm's share construct process, *n* masks will be generated using the mask generation algorithm for *n* individual shares. Individual masks will be ANDed with the Secret to generate the masks. In the Mask generation algorithm the arrangement of rows of size *n*, a matrix will be created with the dimension ${}^{n}C_{k-1}$ x *n*. Such rows will have (*k-1*) the number of 0, and *(n-k+1)* number of 1. By transposing the originally generated matrix, a new matrix will be generated. This matrix's dimension is *n* x ${}^{n}C_{k-1}$. For n shares, every row of the matrix forms the individual mask. Every mask has the size of bits ${}^{n}C_{k-1}$. This algorithm uses the operation ANDing and ORing, which offers the benefit of less computational complexity. It requires cover and it cannot be used for real-time communications.

Socek D *et al.* [14] proposed a scheme that was based on different principles than that used by Y. Desmedt *et al.* [13] and Yang CN. [9]. In place of using sound/wave interference, two types of frequencies short beep and long beep are used, which is quite similar to the Morse code signal. It also reflects strong analogy with visual cryptography as the short beep correspond white pixel, which is mathematically 0, and long beep represents the black pixel which is mathematically 1. For representing the beeps, dashes and dots have been used. Dash (-) represents long beep is, dot (.) represents short beep. Morse code doesn't fall in the category of prefix code as it needs a pause to distinguish between symbols. So a prefix code is used, possibly Huffman encoding scheme. The audio signal that is represented using long and short beep is referred to as Prefix Binary Code (PBC) which act as a simple binary message. For randomly generating the shares coin is tossed, and values assigned to the shares is based on H or T. In this scheme, no cover audio is required, and it is also easy to implement. It is only suitable for binary secrets, cannot be applied to audio secret.

Jing-Zhang *et al.* [15] proposed an audio secret sharing scheme based in which the size of the original secret is reduced. The encoding of the secret audio is based on fractal theory. The fractal theory identifies the most similar part in the audio secret and performs encoding using affine transform. The fractal codes are divided into n parts which are hidden/embedded into audio using least significant bit technique in the frequency domain. Even it reduces the size of the original audio, the total size of the shares is larger than the secret audio. The computation time required is also very high, and it cannot be used in real-time scenarios.

S. Vyavahare *et al.* [17] proposed an audio secret sharing scheme which does not require any cover audio and based on matrix projection technique. In this scheme, frequency samples for input audio are calculated. Audio data is represented in square matrix form, and padding zeroes are added if required. For generating shares, Li Bai`s [23] reliable secret scheme is applied. Here generated shares are in matrix form. These are converted into sequential data shares. By the help of frequency

samples, sequential shares are converted into audio shares. In this scheme, the total size of the shares generated is approximately equal to the secret audio. A remainder matrix is generated during the construction and which is later used for the reconstruction of the secret. If any sort of change occurs in the remainder matrix secret cannot be reconstructed. Remainder matrix comes to a single point of failure which find out to be major drawback of this approach.

A (2, *n*) ASS was proposed by Md. Ehdaie *et al.* [11] based on the properties of the amplitude of the audio signal. It was also later generalized to (*k, n*) ASS. This was the first Audio Secret Sharing schemes which uses audio as a secret. The shares are generated from the secret, so no cover is required. The first share is generated randomly, and all the other shares depend on it. Audio can be represented as a *1 X N* vector matrix consisting amplitude of the audio. This scheme was based on the amplitude property that if multiplication or division is performed with all the elements of P_i (Amplitude vector of an audio) by a constant number *c*, it amplifies or reduces the A_i with amplitude *c*. It means the volume of the sound will increase or decrease depending on the operation performed and its scale will be proportional to *c*. On the basis of the above facts if we consider an audio file possessing amplitude vector P_i and generate Q_i and R_i such that $\forall i : Q_i + R_i = a\ P_i$, *a* is a constant. By performing addition between all the elements of the amplitude vector of Q_i and R_i and then playing the resultant or by playing Q_i and R_i together, we can hear the original sound. In this scheme the total size of the share is larger than the original audio secret but at the same time is it significantly low as compared to other schemes [13, 9, and 4]. If in the randomly generated matrix, a singular sub-matrix occur than we need to recreate the sub-matrix that certainly increases the computation. This scheme is also not applicable to real-time audio.

S.S. Bharti *et al.* [18] proposed a more secure and efficient ASS which can also be applied to real-time audio systems. This method offers three steps for producing verifiable audio shares. The first step transforms audio secrets input into two separate streams: Scaled amplitudes stream and Signs stream. Phase two generates *n* primary shares. These primary shares are created by changing the position of the scaled amplitudes and the relative order of signs. When played primary shares give the illusion of noise. Share number and scaling factor are also included in the primary shares which are needed to later reveal the information. In the third phase, *n* authentication keys are generated and embedded into respective primary shares which are later used to check share integrity. The shares thus created are verifiable. When all the shares have been obtained, then the secret is exposed by conducting some computation. The integrity of the share/s can be checked at the receiver end. This method provides an additional unprecedented feature, i.e., the shares generated are verifiable, which provides a robust method for checking the integrity of the shares if any intruder attacked the share during transmission. It is more secure because even after getting all the shares, no clue of the secret can be obtained because the reconstruction algorithm is essential. The demerit of this scheme is that the reconstructed audio may lose significant-quality due to the existence of scaling factor, i.e., the scheme is not lossless. The correlation coefficient also doesn't remain intact and decrease to 0.99 from 1.

In the previously proposed schemes which are discussed, no scheme offers an *(k, n)* ASS which is secure and possess a robust verification method and also provide quality assurance. In this paper, a Verifiable *(k, n)* Audio Secret Sharing (VASS) scheme has been proposed, which possess a threshold *k* for reconstruction and which guarantees no degradation in quality and intact correlation coefficient. Other limitations found in the literature review are also reduced in the proposed scheme.

3. PROPOSED METHOD FOR AUDIO SECRET SHARING

Flow chart of the proposed approach for generating *n* verifiable shares is shown in figure 1.

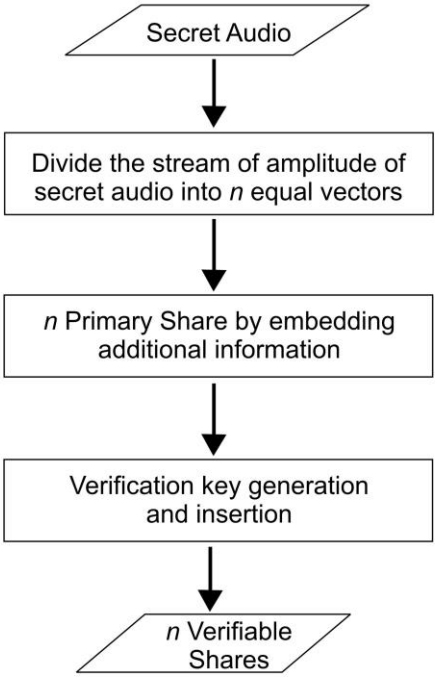

Figure 1. Generating *n* verifiable share

The generation of verifiable shares consists of two phases. In the first phase, the stream of the amplitude of secret audio is divided into *n* equal vectors. These vectors are further used to generate *n* primary shares in the second phase by circular shift operation to insert certain

amplitude from these vectors at a certain location in vectors itself.

Also, in this phase, in all primary shares share number, number of shares *n* and threshold *k* are also inserted, these are essential in the reconstructing the audio secret. Phase two is responsible for making the shares verifiable. A particular verification key is generated for each primary share and is inserted in the respective primary share. This verification key is regenerated by performing some computation at the receivers' end and used for the integrity verification of the share.

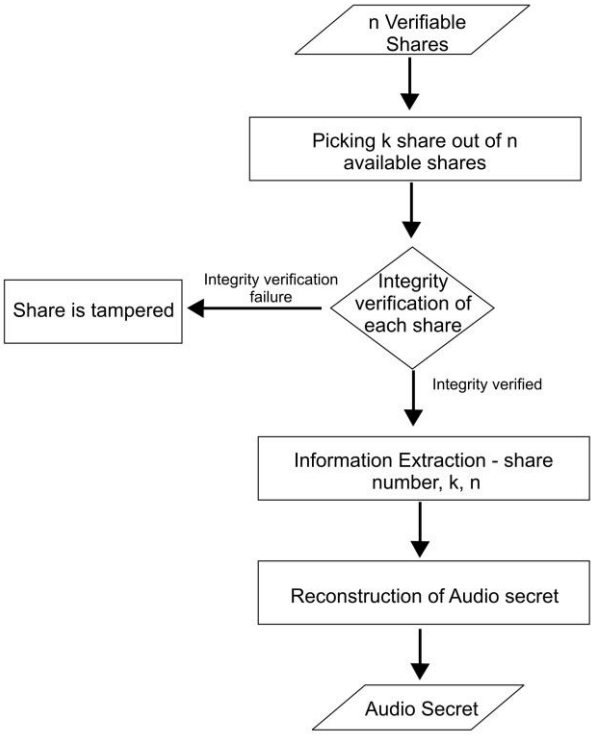

Figure 2. Reconstruction of Secret Audio

Figure 2 shows the flow chart of reconstruction of secret for any input audio secret.

Share number, number of shares *n*, and threshold *k* are being extracted from the shares received. The integrity of each share is verified by regenerating the authentication key. It's also been determined that the number of shares available at the instance of reconstruction is equal to or greater than threshold, if not reconstruction is not possible. For the reconstruction of an audio secret reverse circular shift operation is performed.

A. *Divide the stream of audio into n equal vectors*

For the context of this paper $sh^1, sh^2, \ldots, sh^n$ represents a share in the vector space or temporary shares, i.e., sh^i represent i^{th} temporary share and $shp^1, shp^2, \ldots, shp^n$ represents primary shares, i.e., shp^i represent i^{th} primary share. Stream of the audio amplitude of size *N X 1* is divided into *n* vectors of equal size by applying *modulus n* operator on indices, where *N/n X 1* is the size of each vector and *n* vectors $sh^1, sh^2, \ldots, sh^n$ are generated for generating *n* primary shares $shp^1, shp^2, \ldots, shp^n$.

Algorithm 1: *Algorithm for dividing the stream of audio into n equal vectors*
Input: *O^A : Original Audio Stream of Amplitudes of size NX1, n : number of shares to be generated, k : minimum threshold for recovery*
Output: *$sh^1, sh^2, \ldots, sh^n$: n Vectors or temporary shares of each size N/n*
share matrix : Vector Matrix used for share generation

```
1: index = 1;
2: N = size(O^A)     /* length of  original audio */
3: N/n = len;                /* length of each share */
4: for i = 1 : len do
5:       for j = 1 : n do
6:           share matrix[i][j] = O^A(index)
7:           index = index + 1
8:       end for
9: end for
```

Let`s consider an example of an audio stream consisting amplitudes = [27, 165, 32, 15, 207, 35, 146, 52, 17, 23, 104, 43] and *N = 12* and *n = 4* and *k = 2*. Hence 4 vectors are created of N/n size each that is 3.

TABLE I. GENERATED VECTORS FROM AUDIO STREAM

sh^1	sh^2	sh^3	sh^4
27	165	32	15
207	35	146	52
17	23	104	43

B. *Generating n Primary Shares and making share verifiable.*

Here we will be using data block, which can be termed as a set of elements that consists of *N* elements and possess a dimension of *(N/n) X n.* Round is equal to the difference between the number of shares to be generated *n* and threshold *k* (which is 2 in the above example) which signifies that how many blocks of data is needed to be inserted in the vector space to make primary shares and in each vector *(N/n)* elements are inserted for each round. If round equals to 2, then two data blocks consisting of *2N* elements need to be inserted. Making each vector size from *(N/n) to 3(N/n).*

Algorithm 2: *Algorithm for n verifiable share generation*
Input: *share matrix : Vector space possessing $sh^1, sh^2 \ldots sh^n$ which are used for generating primary shares $shp^1, shp^2 \ldots shp^n$, n : number of shares to be generated, k : threshold for recovery*
Output: *$shp^1, shp^2 \ldots shp^n$ n shares*

```
1: index = 1;
```

2: N = size(O^A)
3: N/n = len; / length of each share */*
4: ***for*** *round = 1 : (n − k)* ***do***
5: inc = round
6: ***for*** *i = (round ∗ len) + 1 : (round + 1)len* ***do***
7: ***for*** *j = 1 : n* ***do***
8: col = j + inc
9: ***if*** *(col > n)* ***then***
10: col = col − n
11: ***end if***
12: share matrix[i][j] = share matrix[i − (round ∗ len)][col]
13: ***end for***
14: inc = inc + 1
15: ***if*** *(inc == n)* ***then***
16: inc = 1
17: ***end if***
18: ***end for***
19: ***end for***
20: ***for*** *i = 1 : n* ***do***
21: $shp^i(3)$ = Average(share matrix[5][i], share matrix[4][i]) + n
22: $shp^i(2)$ = Average(share matrix[5][i], share matrix[5][i], $shp^i(3)$) + k
23: $shp^i(1) = shp^i(2)$ + sharenumber
24: $shp^i(4)$ = share matrix[][i] / Insert complete vector from share matrix from index 4 in shp^i */*
25: end for
26: ***for*** *i = 1 : n* ***do***
27: temp = 0
28: ***for*** *j = 1 : size.shp^i* ***do***
29: temp = temp ⊕ $shp^i[j]$ / Verification key generation */*
30: ***end for***
31: shp^i(size.shp^i + 1) = temp / Embedding Verification key */*
32: ***end for***
33: ***for*** *i = 1 : n* ***do***
34: ***Distribute share*** *Shp^i to i^{th} member*
35: ***end for***

The elements which are inserted to the vectors belong to the vector itself. There are two rules which should be followed while inserting the elements.

Rule 1: Elements from the other vector are inserted to some other vector that means element form one particular vector will not be inserted in the same vector. The insertion of the elements is performed using a circular shift. In a circular shift, we shift the elements right to left, and the left-most element becomes the rightmost. So if we have three elements 1, 2, and 3 and perform one circular shift, then we get 2, 3, and 1. For every round, we will be inserting a data block. The indices range of a data block in a round varies from *(round*(N/n)) + 1 to (N/n (round+1))*. So in the previous placed example, the range of data block for round 1 for each share will be from 4 to 6, i.e., in round 1 insertion will be made in each share from 4 to 6 indices and for round two it will be 7 to 9.

Rule 2: The inserting element at row *i* of some vector will be from *i modulus (N/n)*. It means that the i^{th} *row* of the vector-matrix will have elements from the *i modulus (N/n)* row.

The row will be inserted after performing a certain number of circular shifts. The number of circular shift performed on each row will be different. The number of shift operations performed on the first row of the data block will be equal to the current round, and with each upcoming row, the shift will linearly increase by 1.

As the number of shifts to be performed is incremented by each coming row, there will be a case when circular shifts to be performed will be equal to *n*, at that instance of time, we need to reset the circular shift to 1. If we don`t reset, elements of the same vector will be inserted in the same vector, which will be a violation of rule 1.

TABLE II. AFTER ONE INSERTION OF ROUND ONE

Row no.	sh^1	sh^2	sh^3	sh^4
1	27	165	32	15
2	207	35	146	52
3	17	23	104	43
4	165	32	15	27

In the example, the number of circular shifts to be performed before inserting in round 1 for 4, 5, and 6 rows will be 1, 2, and 3 respectively, and for round 2 for row 7, 8 and 9 will be 2, 3 and 1 respectively. Insertion in 4^{th} row will be from *4%(N/n)*, which is *4%3 = 1^{st}* row with 1 circular shift.

TABLE III. AFTER ROUND ONE OF INSERTION

sh^1	sh^2	sh^3	sh^4
27	165	32	15
207	35	146	52
17	23	104	43
165	32	15	27
146	52	207	35
43	17	23	104

TABLE IV. AFTER ROUND TWO OF INSERTION

sh^1	sh^2	sh^3	sh^4
27	165	32	15
207	35	146	52
17	23	104	43
165	32	15	27
146	52	207	35

43	17	23	104
32	15	27	165
52	207	35	146
23	104	43	17

For reconstructing the share, we need the share number, number of shares *n*, and threshold *k*. So the same, we will insert these in these vectors.

For the generation of primary share, the *share number*, number of shares *n*, and threshold *k* are inserted at the top of vector sh^1, sh^2 ...sh^n making them primary share shp^1, shp^2 ...shp^n. The threshold *k* is inserted at index 3 by performing a summation of it with the average of $sh^i(4)$ and $sh^i(5)$. Now for insertion of *n* summation is performed between *n* and average of $sh^i(3)$, $sh^i(4)$ and $sh^i(5)$ and inserted at $sh^i(2)$. For inserting the share, number summation is performed between share number and $sh^i(2)$.

TABLE V. THE FOLLOWING VALUES NEED TO BE INSERTED IN THE RESPECTIVE SHARES AND WILL BE USED DURING RECONSTRUCTION

sh^1	sh^2	sh^3	sh^4
122	105	94	39
121	104	93	38
119	102	91	35

For $sh^1(3)$ = Average(207+27)+*k* = 117+2 = 119. $sh^1(2)$ = Average(207+27+117)+*n* = 121. $sh^1(1)$ = 121 + 1 = 122. For sh^2, sh^3 and sh^4 these values will be (102, 104, 105), (91, 93, 94) and (35, 38, 39) respectively.

Average of elements have been used for this, the reason of choosing average over other operation for inserting the value directly is preferred because averaging cause resemblance which means the elements show resemblance or similarity with next and previous element. If this insertion is performed without using the averaging method can make the element look suspicious as it may stand out in the signal. The authenticity of these primary secret cannot be verified, so for ensuring the integrity of the primary shares verification key used. For making the primary shares verifiable, and first verification key is generated for each primary share and inserted at the end of the primary share. For the generation of the verification key bitwise XOR operation performed on all the elements of the primary share.

For shp^1 : *Bitwise-XOR*(122, 121, 119, 27, 207, 17, 165, 146, 43, 32, 52, 23) = 185
If we consider the shp^1 and perform bitwise XOR between its elements, then we will get 185 and for shp^2, shp^3, and shp^4 will get 91, 70, and 68, respectively. After inserting information which is essential for reconstruction and bitwise XOR, the temporary vectors become verifiable shares which will be sent to respective members (table VI).

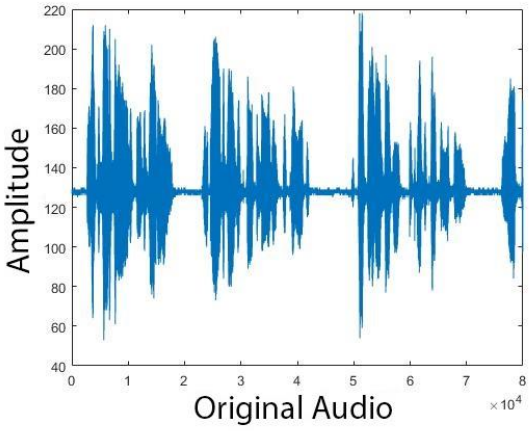

Figure 3. Input Audio Secret

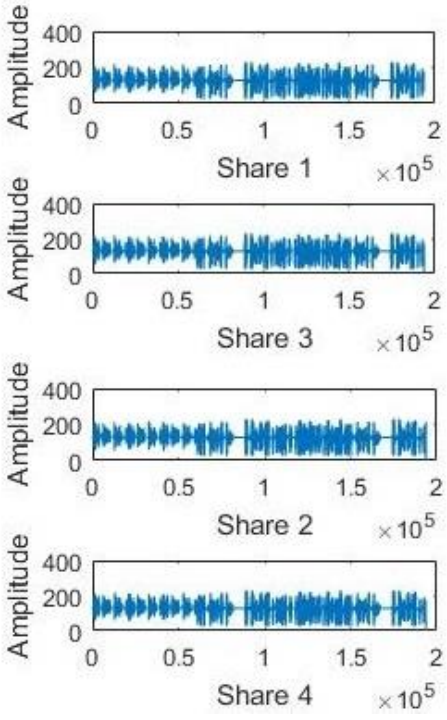

Figure 4. (k, n) Verifiable Primary Shares

TABLE VI. VERIFIABLE SHARES

shp^1	shp^2	shp^3	shp^4
122	105	94	39
121	104	93	38
119	102	91	35
27	165	32	15
207	35	146	52
17	23	104	43
165	32	15	27
146	52	207	35
43	17	23	104

32	15	27	165
52	207	35	146
23	104	43	17
185	91	70	68

C. Reconstruction of the Audio secret

Reconstruction is performed into two phases. First, we extract the *share number*, *n* and *k* form the received secret and determine *N*, then reconstruct the audio secret.

Algorithm 3: *Algorithm for verification of share integrity and information extraction*
Input: *m: number of share shp received*
Output: *Reconstruction possible or not and shares are genuine or tampered*
1: share number= $shp^i(1)$ - shp^i (2)
2: n = Average(shp^i (5) + shp^i (4) + shp^i (3)) - shp^i (2)
3: k = Average(shp^i (5) + shp^i (4)) - shp^i (3)
4: ***if*** *(m < k)* ***then***
5: Reconstruction not possible shares less then threshold
6: ***else***
7: Minimum threshold available reconstruction possible
8: ***end if***
9: temp = 0
10: ***for*** *j 1 : size. shp^i - 1* ***do***
11: temp = temp ⊕ $shp^i[j]$ / Verification key regeneration */*
12: ***end for***
13: ***if*** *(temp == $shp^i(size.shp^i)$)* ***then*** */* Key Verification */*
14: Share is genuine with no tampering
15: ***else***
16: Share is not genuine tampering in share
17: ***end if***

For performing the integrity verification we perform bitwise XOR between all the elements from indices 1 *to* *(size.shp^i – 1)* and compare this to the last element of the share, if they are equal then the share is intact if not then share has been tampered.

Before reconstructing the audio, we perform integrity verification and check whether the number of shares received is equivalent to or greater than threshold *k* or not. If not, then secret audio reconstruction is not possible.

In the framed example (table VII), *n* = 4 number of shares are available and will be randomly picking threshold *k* = 2 of them and will verify their integrity, extract information and perform the reconstruction. Until we don`t know the actual share numbers, we will be calling them as unknown share 1 *(US^1)* and unknown share 2 *(US^2)*.

TABLE VII. 2 SHARES PICKED FOR RECONSTRUCTION

US^1	US^2
122	94
121	93
119	91
27	32
207	146
17	104
165	15
146	207
43	23
32	27
52	35
23	43
185	70

For these unknown shares US^1 and US^2 we need to determine *share number*, *n*, and *k*. For US^1 *k = 119 - Average(207,27)* which will be 2, *n = 121 – Average(107,27,119)* which will be 4 and *share number = 122 – 121* which will be 1. Hence US^1 is *share number* 1. Similarly, for US^2 *k* will be 2, *n* will be 4, and *share number* will be 3.

The number of rounds can be determined by *n – k* and *N* can be determined by *((size.shp^i -4) % (round +1))* we are subtracting 4 because we have inserted additional information at 4 places. As we know from the share construction process that the inserted element at the location *i* of some vector will be from location *i modulus (N/n)*. It also signifies that the i^{th} *row* of the vector-matrix will have elements from the *i modulus (N/n)* row, and we perform a circular shift to do the same. For reconstruction, we will perform a reverse circular shift, i.e., shifting elements left to right in a circular fashion. We will be performing a reverse circular shift on each data block of each round and then pick the elements for reconstruction.

First of all, we will arrange all the share in the reconstruction vector space and will remove the additional elements from all the available share, i.e., the last element, which is the verification key, and the three elements from the beginning because they are extrinsically calculated and inserted for information transmission.

Now will be arranging these share in the increasing order of their share number, leaving the vector blank for missing share.

NULL or -1 or any other element (which is not the part of the amplitude space we are dealing with) is inserted at all the blank spaces present in the vector. Here we are using -1 because we don't have any negative values in our amplitude space.

TABLE VIII. RECONSTRUCTION SPACE AFTER INSERTING AVALIABLE SHARES AND -1 IN EMPTY SHARES

shp^1	shp^2	shp^3	shp^4
27	- 1	32	- 1
207	- 1	146	- 1
17	- 1	104	- 1
165	- 1	15	- 1
146	- 1	207	- 1
43	- 1	23	- 1
32	- 1	27	- 1
52	- 1	35	- 1
23	- 1	43	- 1

Now reverse circular shift will be performed form *(round*(N/n) + 1) to (N/n)(round + 1)* for each round. The number of reverse shifts to be performed at a particular row *i* of this reconstruction vector space will depend on the value of round. The number of reverse circular shifts performed on each row will be different. The number of reverse shift operation performed on the first row of the data block of a particular round will be equal to the current round, and with each upcoming row, the reverse shift will linearly increase by 1. As the number of reverse shifts to be performed incremented by each coming row, there will be a case when reverse circular shifts to be performed will be equal to n. At that instance of time, we need to reset the circular shift to 1.

TABLE IX. RECONSTRUCTION SPACE AFTER REVERSE CIRCUALR SHIFT

shp^1	shp^2	shp^3	shp^4
27	- 1	32	- 1
207	- 1	146	- 1
17	- 1	104	- 1
-1	165	-1	15
207	- 1	146	- 1
-1	23	-1	43
27	- 1	32	- 1
-1	35	-1	52
-1	23	-1	43

Now, the value of any index of $shp^i(index)$, index lying between *1 to N/n* can be picked from *index + round*(N/n)* for a particular round if the element at *index + round*(N/n)* is -1 then round value should be incremented and then the value can be picked form next corresponding location. For $shp^2(1)$, its value will be picked form (1 + 1*3) or (1 + 2*3). Hence it is available at the first location so don't need to access the second location. In case of $shp^2(2)$, we will get -1 at the first location so second location need to be access for getting value of shp^2. In the same manner all the elements are retrieved from 1 to *N/n* for each share and the secret audio is reconstructed.

Algorithm 4: *Algorithm for reconstructing audio secret*
Input: *shp^1, shp^2...shp^k shares: k shares, N: Length of original audio, n: number of shares, k: threshold*
Output: *O^A : Reconstructed audio stream*
recon : *reconstruction vector space of size (round+1)N/n X n, all the available shares are inserted in order of their share number and empty shares are filled with -1.*

```
1: N/n = len;        /* length of each share */
2: for round = 1 : (n − k) do
3:     inc = round    /* inc signifies how many
                       shift to perform */
4:     for i = (round ∗ len) + 1 : (round + 1)len do
5:         for j = n:2 do
6:                 temp = recon[i][n]
7:                 recon[i][n] = recon[i][n-1]
8:                 if (j == 2) then
9:                         recon[i][1] = temp
10:                end if
11:        end for
12:  end for
13: inc = inc + 1
14:  if (inc == n) then
15:      inc = 1
16: end if
17: end for
18: for j = 1:n do
19:   for i = 1:(N/n) do
20:     for z = 1: n-k do
21:         if (recon[i][j]  == -1) then
22:             recon[i][j] = recon[i+round*len][j]
23:         else
24:           break
25:         end if
26:      end for
27:   end for
28: end for
29: k=1
30: for i = 1:len do
31:         for j = 1:n do
30:           O^A(k) = recon[i][j]
31:           k=k+1
32:         end for
33: end for
```

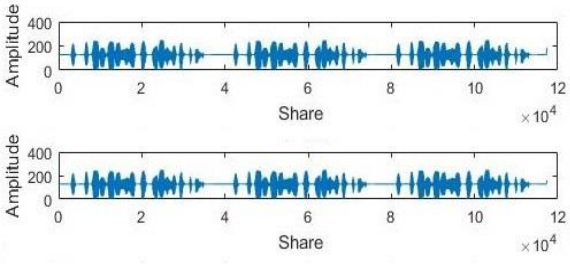

Figure 5. Two received shares for reconstruction

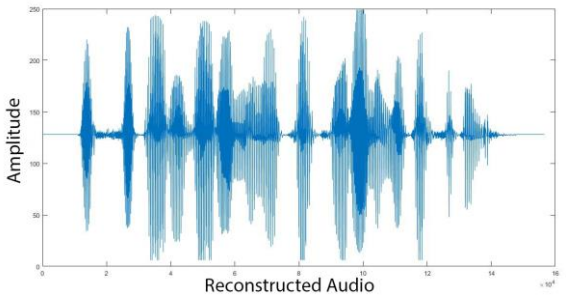

Figure 6. Reconstructed Audio Secret

4. Analysis of Experimental Results

The most popular objective/subjective parameters which are used to measure the quality of the audio signal are MOS [10], CMOS [7, 10], correlation coefficient (r) , similarity factor (S_f) and PSEQ.

Additionally, a security analysis has also been performed in which genuine shares are corrupted, and their integrity verification is performed to show the robustness of the proposed verification method.

A. Database used for experiment

The audio secrets are taken from IndicTTS [2] database having various durations. IndicTTS database contains male and female pronounced sentences possess in 13 different Indian languages.

B. Performance Measurements

The performance of the proposed algorithm will be evaluated by performing a comparative analysis of the original and reconstructed audio on various parameters such as quality, similarity and correlation. Objective as well as subjective techniques are used for performance measurement. We are using Comparison Mean Opinion Score (CMOS) [7, 10], Mean Opinion Score (MOS) [10], Correlation Coefficient, Similarity Factor and PESQ (Perceptual Evaluation of Speech Quality) [24-25].

For calculating the CMOS and MOS, which are subjective parameters, a group of 10 persons is considered. In this group, original audio and reconstructed audio have been played. For CMOS they are asked to compare the quality of the first audio (original) with second audio (reconstructed) and rate according to the labels in table X. Here we are using seven labels. Hence rating received will be on the scale of 3 to -3 and the average of these rating will be calculated.

TABLE X. Labels used for CMOS [7, 10]

Rating	*Label*
3	Very Good
2	Good
1	Slightly Good
0	About the same
-1	Slightly worse
-2	Worse
-3	Much Worse

For MOS persons in the group are asked to provide the rating for audio quality of the reconstructed secret based on the predefined labels individually for both the reconstructed audios. Predefined labels for MOS are defined in table XI. Here we are using five labels. Hence rating received will be on the scale of 5 to 1 and the average of these rating will be calculated.

Correlation coefficient establish a correlation between the original secret and its reconstructed version. It varies from 1.0 to -1.0.

Similarity factor (S_f) is percentage scale that is based on the amplitude vector of the audio. This scale compares all the audio amplitudes from the original audio amplitude vector and to the reconstructed audio vector in sequential order. It maintains total number of amplitudes compared and total number of amplitudes which are similar in both audios. S_f is calculated as follows –

$$S_f = (No.\ of\ Similar\ Amplitude/Total\ Amplitude)\ *100$$

Let assume the original audio amplitude vector OA = [10, 20, 30, 40, 50] and reconstructed audio amplitude vector RA = [10, 20, 30, 40, 50] in this case S_f comes out to be 100% , showing both the amplitude vectors are exactly the same. In another scenario let reconstructed audio amplitude vector RA = [10, 20, 56, 30, 50] in this scenario S_f comes out to be 75%, signifying that 75% amplitude in both the audios are similar with respect to amplitude and position in the amplitude vector.

PESQ - Perceptual Evaluation of Speech Quality (Raw MOS, MOS-LQO) which is an ITU standard is also been evaluated for both the original and reconstructed audio. Raw MOS range between -0.5 to 4.5 and MOS-LQO range between 1 to 5.

MOS, CMOS and PESQ are majorly quality measures that analyze the quality of the reconstructed audio and compare it with the original audio. They make sure that the audio quality didn't degraded during the share generation and reconstruction phase and in transmission as well. Correlation Coefficient and Similarity factor (S_f) analyze and make sure that the reconstructed audio is the same as original audio in context and volume. Any iteration in it will certainly make the correlation coefficient less than 1 and Similarity factor (S_f) less than 100.

TABLE XI. Labels used for MOS [10]

Rating	*Label*
5	Excellent
4	Good
3	Fair
2	Poor
1	Bad

C. Experiment Results

For the input audio signal for a male shown in figure 7, figure 8 shows four created verifiable shares, and figure 9 shows the reconstructed audio secret.

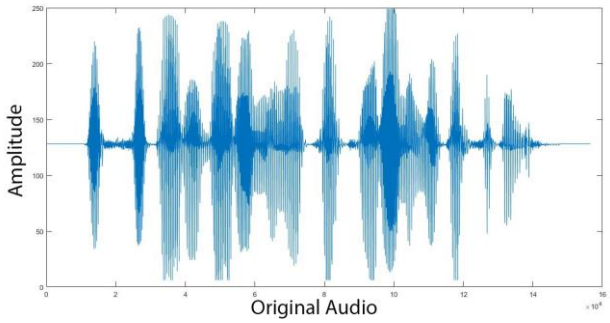

Figure 7. Input Audio Secret (Male)

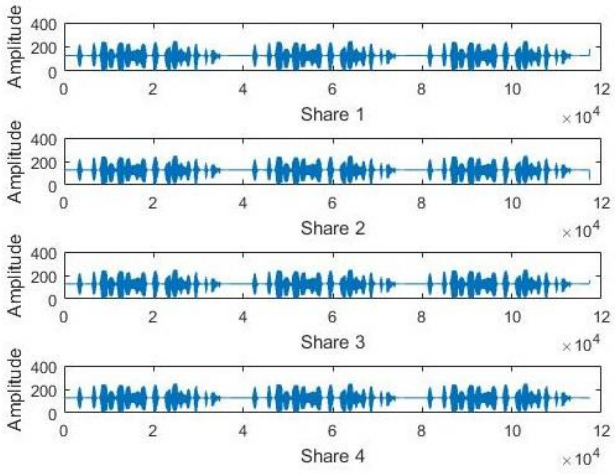

Figure 8. Shares generated

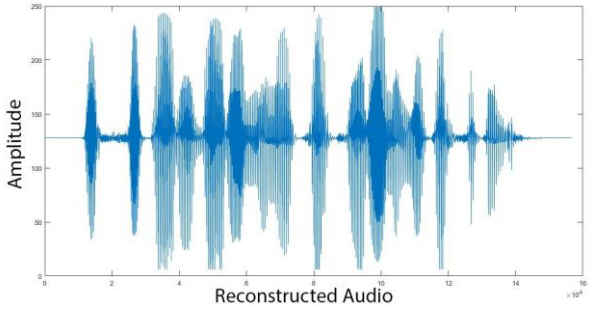

Figure 9. Reconstructed Audio Secret (Male)

Figure 10 shows another example of an input audio secret, which is a female voice. Figure 11 shows four shares created, and figure 12 shows the reconstructed audio secret.

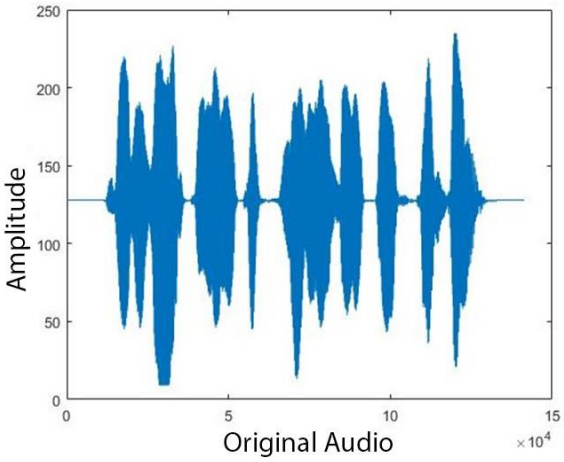

Figure 10. Input Secret Audio (Female)

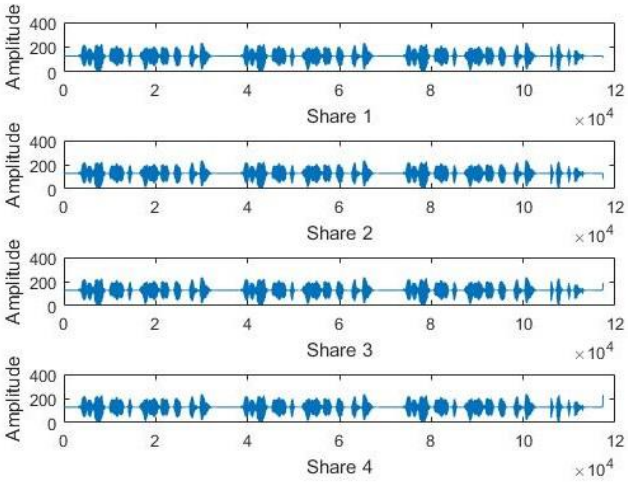

Figure 11. Shares generated

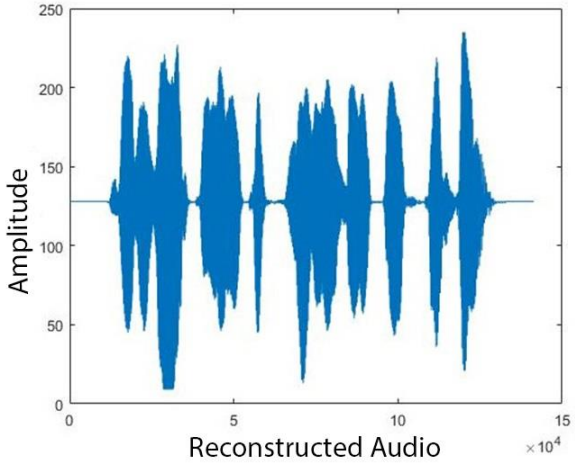

Figure 12. Reconstructed Audio Secret (Female)

For both the audio secret Comparison Mean Opinion Score (CMOS) [7, 10], Mean Opinion Score (MOS) [10], Correlation Coefficient, Similarity Factor and PESQ (Perceptual Evaluation of Speech Quality) [24-25] is calculated and further analyzed.

The Correlation Coefficient and Similarity Factor (S_f) has been calculated for both these cases and which comes out to be 1 and 100% respectively for both the cases which shows lossless trait of the algorithm. This also shows the intact nature of the algorithm. The original secret audio and the reconstructed secret audio are exactly the same.

PESQ (Raw MOS, MOS-LQO) comes to 4.500 and 4.549 for both male audio (figure 13) and female audio (figure 14) respectively that shows the audio didn't degraded and possess the same quality.

```
Reading reference file male_original.wav...done.
Reading degraded file male_reconstructed.wav...done.
 Level normalization...
 IRS filtering...
 Variable delay compensation...
 Acoustic model processing...

P.862 Prediction (Raw MOS, MOS-LQO):  = 4.500   4.549
```

Figure 13. PESQ test for Input Audio Secret (Male)

```
Reading reference file female_original.wav...done.
Reading degraded file female_reconstructed.wav...done.
 Level normalization...
 IRS filtering...
 Variable delay compensation...
 Acoustic model processing...

P.862 Prediction (Raw MOS, MOS-LQO):  = 4.500   4.549
```

Figure 14. PESQ test for Input Audio Secret (Fenale)

The MOS readings has been taken for these two reconstructed audios (table XII), and its value found to be 4.8 for the first audio secret and 4.9 for the second audio. This shows the quality of the audio secret does not diminish, and it is the same as the original audio.

TABLE XII. MOS RATING FOR BOTH RECONSTRUCTED AUDIO SECRET

Person in group	Rating for Reconstructed Male audio secret	Rating for Reconstructed Female audio secret
1	5	5
2	5	4
3	5	5
4	5	5
5	5	5
6	3	5
7	5	5
8	5	5
9	5	5
10	5	5
Average	4.8	4.9

TABLE XIII. CMOS RATING FOR BOTH AUDIO

Person in group	Rating for Input 1, male audio secret	Rating for Input 1, Female audio secret
1	0	0
2	0	0
3	1	0
4	0	0
5	0	0
6	0	0
7	0	0
8	0	0
9	0	0
10	-1	1
Average	0	0.01

CMOS (table XIII) comes out to be 0 and 0.01 which came out to be "About the same" when referenced to the CMOS rating labels. It shows that both the original and reconstructed audio are nearly the same in quality.

All these measures signifies the intact nature of the approach and the lossless transmission of amplitude form original audio to shares and share to reconstructed audio.

D. Security Analysis

There may be possible cases in which an intruder tries to manipulate or corrupt the amplitude of the shares. Until the number of corrupted shares is less than *n-k*, there will be no issue in reconstruction. For each share, a verification key is generated and inserted in the share, and at the receiver end, it is regenerated and matched. Our Scheme is able to identify the share which is tampered.

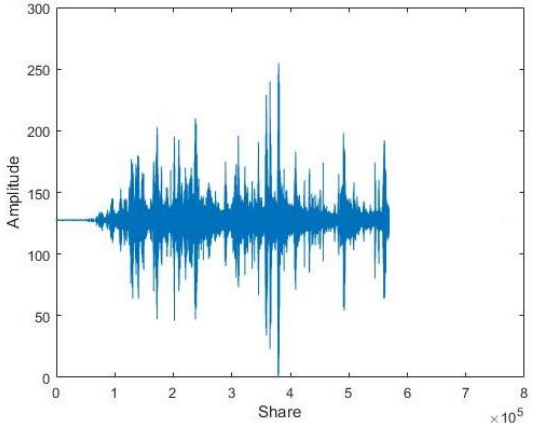

Figure 15. Genuine Share

For this experiment has been performed in two cases:

Case 1: Only few amplitude elements have been altered, so the shares looks similar.

Case 2: A random noise is added throughout the share

Only three amplitude values have been altered in the genuine share (figure 15) and then on the tampered (figure 16) share verification is performed.

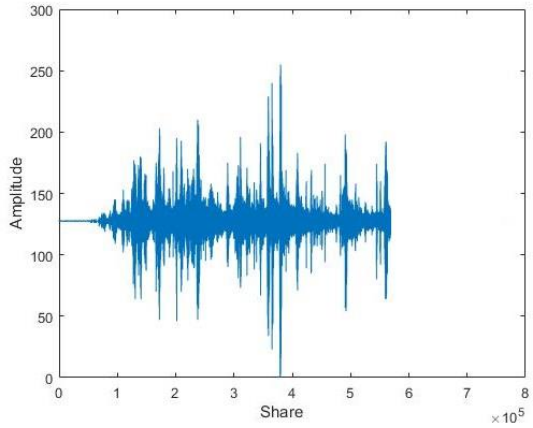

Figure 16. Tampered Share

Both of the shares are looking exactly the same, but in genuine share, the bitwise-XOR value is 113, which is equivalent to the verification key stored in the share, and in the tampered share, bitwise XOR values come to 205, which is not equal to the verification key hence this share has been tampered.

Figure 17 shows another genuine share, and Figure 18 shows genuine share tampered by adding random noises to amplitude.

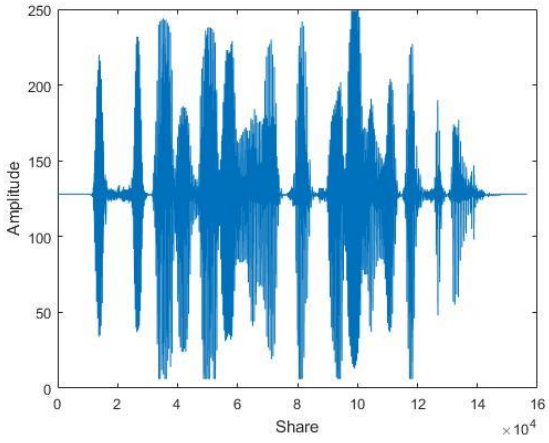

Figure 17. Genuine Share

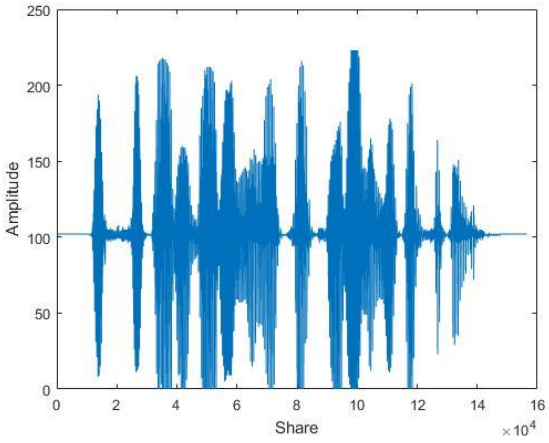

Figure 18. Tampered Share by adding random noise

In this case, XOR for genuine share was 96 and is equivalent to verification key, and for tampered share XOR is 147 that is not equal to the verification key, even in this case, random addition in amplitude also corrupted the key.

E. Comparion of proposed algorithm with existing

For comparing this proposed approach with other approaches, different criteria have been taken into account, as shown in the table XIV. The table XIV consists of the following parameters –

1) k Threshold: This is the threshold, the minimum number of shares required for the regeneration of the audio secret.

2) Lossless: This determines that the reconstructed secret loses some data or precision. If yes, it can degrade the audio quality as well.

3) Correlation Coefficient: This establishes a correlation between the original secret and its reconstructed version. If the scheme is intact in quality and possesses no loss, then it should be 1.

4) Cover Required: Cover is the audio(or other media) in which the secret is embedded. It signifies that cover is required or not.

5) MOS: Mean Opinion Score is a subjective parameter used for determining the quality.

6) Share Verification: Method of verifying the integrity of the received share.

TABLE XIV. COMPARISON BETWEEN PROPOSED SCHEME AND RELATED STATE OF THE ARTS

Scheme	k - Threshold	Lossless	Correlation Coefficient = 1	Cover Require	MOS	Share Verification
Y. Desmedt *et al.* [13]	No	-	-	Yes	-	No
Ching-Nung Yang [9]	No	-	-	Yes	-	No
Prabir *et al.* [8]	No	-	-	Yes	-	No
Socek D *et al.* [14]	No	-	-	Yes	-	No
Md. Ehdaie *et al.* [11]	Yes	-	-	No	-	No
Chen-chi-Lin *et al.* [4]	No	-	-	Yes	-	No
S. Vyavahare *et al.* [17]	Yes	Yes	Yes	No	-	No
Jing-Zhang *et al.*[15]	No	No	No	Yes	-	No
S.S. Bharti *et al.* [18]	No	No	No	No	3.8 - 4	Yes
Proposed	Yes	Yes	Yes	No	4.8 – 4.9	Yes

5. CONCLUSION

This scheme can be used for communication channels and secure transmission of audio. It can also be used in VoIP [22] methodology to provide voice communication protection. It is *(k, n)* Audio Secret Sharing where only *k* number of shares are required to reconstruct the lossless secret audio. The proposed scheme has the ability to authenticate the shares that are genuine or not, and identified the share numbers, which are tempered. In this scheme, no need of cover to transmit the share. Correlation Coefficient remains intact in this scheme and equal to 1 and Similarity Factor (S_f) comes out be 100% which shows the lossless recovery of the audio secret. MOS comes to be between 4.8 and 4.9 on the scale of 5, CMOS comes to be 0 and 0.01 and PESQ (Raw MOS, MOS-LQO) comes to 4.500 and 4.549 for both audio secret which signifies excellent quality of the reconstructed secret audio. In this scheme, the total size of all the shares is larger than the original audio that opens up the scope of further research and improvement.

REFERENCES

[1] Blakley GR (1979) Safeguarding cryptographic keys. In: Proceedings of americian federation of information processing societies national computer conference, (AFIPS'79), California, pp 313– 317

[2] Baby A, Thomas AL, Nishanthi NL, Consortium T Resources for Indian languages. In: CBBLR – Community Based Building of Language Resources, Sep 2016, pp 37–43, Brno, Czech Republic: Tribun EU. [Online] Available: https://www.iitm.ac.in/donlab/tts/index.php

[3] Shamir A (1979) How to share a secret. Commun ACM 22(11):612–613

[4] Lin C-C, Laih C-S, Yang C-N (2003) New audio secret sharing schemes with time division technique. J Inf Sci Eng 19(4):605–614

[5] Shannon, C.E.: Communication theory of secrecy systems. Bell Systems Technical Journal, 28 (1949), 656-715

[6] Ehdaie M, Eghlidos T, Aref MR (2008) A novel secret sharing scheme from audio perspective. In: Proceedings of international symposium on telecommunications (IST2008). IEEE, pp 13–18

[7] ITU-T, Subjective performance assessment of telephone band and wide-band digital codecs, ITU-T Recommendation p. 830 (1996)

[8] Prabir Kr. Naskar, Hari Narayan Khan, Ujjal Roy, Ayan Chaudhuri, Atal Chaudhuri, "Shared Cryptography with Embedded Session Key for Secret Audio", International Journal of Computer Applications (0975 – 8887) Volume 26– No.8, July 2011

[9]Yang CN (2002) Improvements on audio and optical cryptography. J Inf Sci Eng 18(3):381–391

[10] International Telecommunication Union - Radiocommunication Sector, Recommendation BS. 562-3, Subjective assessment of sound quality (1990)

[11] Mohammad E, Taraneh E, Reza AM (2008) Some new issues on secret sharing schemes. In: Int'l conference on telecommunications (ICT' 08), June 16–19, St. Petersburg, Russia

[12] Vernam, G.S.: Secret signaling system. U.S. Patent # 1,310,719, 22 Jul 1919

[13] Desmedt Y, Hou S, Quisquater J-J (1998) Audio and optical cryptography. In: ASIACRYPT'98, LNCS, vol 1514, pp 392–404

[14] Socek D, Magliveras SS (2005) General access structures in audio cryptography. In: Proceedings ofIEEE international conference on electro information technology, p 6

[15] Wang JZ, Wu TX, Sun TY (2015) An audio secret sharing system based on fractal encoding. In: Proceedings of 49th, international Carnahan conference on security technology (ICCST), pp 211– 216

[16] NishimuraNorihiro R, Suzuki F (2005) Y suzuki audio secret sharing for 1-Bit audio. Lect Notes Comput Sci 3682:1152–1158

[17] Vyavahare S, Patil S (2016) Analysing secret sharing schemes for audio sharing. Int J Comput Appl 137(11):39–42

[18] Bharti, S.S., Gupta, M. & Agarwal, S. A novel approach for verifiable (n, n) audio secret sharing scheme. Multimed Tools Appl 77, 25629–25657 (2018).

[19] Naor, M.: Visual cryptography. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 1–12. Springer, Heidelberg (1995)

[20] Mainejar Yadav and Ranvijay, "Verifiable Essential Secret Image Sharing with Multiple Decryption" in International Journal of Recent Technology and Engineering (2019).

[21] Mainejar Yadav and Ranvijay, "Collusion attacks in XOR-based RG-VSS". in International Journal of Innovative Technology and Exploring Engineering (2019).

[22] Ryouichi Nishimura, Shun-ichiro Abe, Norihiro Fujita and Yoiti Suzuki, "Reinforcement of VoIP Security with Multipath Routing and Secret Sharing Scheme", Journal of Information Hiding and Multimedia Signal, Volume 1, Number 3, July 2010

[23] Li Bai, "A Strong Ramp Secret Sharing Scheme Using Matrix Projection", IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, 2006.

[24] ITU, Perceptual evaluation of speech quality (PESQ), and objective method for endto-end speech quality assessment of narrowband telephone networks and speech codecs. ITU-T Recommendation p. 862 (2000)

[25] Rix, A., Beerends, J., Hollier, M., Hekstra, A.: Perceptual evaluation of speech quality (PESQ) - A new method for speech quality assessment of telephone networks and codecs. In: Proc. IEEE Int. Conf. Acoust, Speech, Signal Processing, vol. 2, pp. 749–752 (2001)

Sachin Kumar Singh is a Computer Science graduate from Rajkiya Engineering College Sonbhadra, affiliated to Dr. A.P.J. Abdul Kalam Technical University, Lucknow, Uttar Pradesh, India. His current research interest includes Multimedia Security, Data Driven Computation and Data Visualization.

Mainejar Yadav received an M.Tech degree from MNNIT, Allahabad, and presently pursuing a Ph.D. from MNNIT, Allahabad. He is an Assistant Professor of the Computer Science and Engineering Department at Rajkiya Engineering College, Sonbhadra. His areas of interest include visual cryptography, audio secret sharing, digital watermarking, and network security.