Autherizing Audio Remote Access using Encoded Hidden Data

Asst.Prof .Dr. Sana Ahmed Kadhim Asst.Prof .Dr. Saad Abdual Azize
Computer science dept, Al_M'amoon University College

Abstract
Remote access to data or special applications became essential part of the internet applications for fast and secure data usage. Accessing data using audio has been used frequently but audio needs to be verified. In this paper, authorizing audio is done by hiding identification information within the audio. The hidden information is encoded using Huffman tree, encrypted using knapsack algorithm and hidden in specific parts of the audio. The suggested method was tested for corruption and quality of the authentic audio using subjective and some objective test metrics.

Key words Huffman encoding tree, secrete and public key, digital audio, selective hiding, Knapsack encryption, zigzag method, audio transformation.

I. INTRODUCTION

The rapid manner of life nowadays extracted the need for instant accessing of data no matter where and when. Remote access to information using Internet is one of the most usable applications especially in business, politics and news broadcasting. Information to be remotely accessed should be kept secure and integrated by authorizing the access and verifying the means of accessing to these data [1]. Audio has been used recently in remote access to database, banks accounts, personal and work computers, houses and even some confidential applications [2]. Due to the massive improvements in the methods of intruding and stealing, improved securing method had to be invented. Audio alone becomes insufficient to prove access authorization therefore another level of security should be added to audio as a method of remote accessing [3]. The sensitivity of audio nature makes it very difficult to be used as a covered media[4]. Noise and distortion may be occurred when modifying audio. Therefore, the integrity of audio file that is used as a cover should be kept. Data to be included in the audio should be confidential so that it is known only by the owner whose going to use them in remote access to his (her) data (or application, account,…). Cryptography is used to encrypt confidential data, knapsack is a public key cryptosystem which is going to be used in the proposed method for both encryption and selecting hiding positions [5]. Stego_technique may be considered an appropriate technique if it achieves the following: first, using the maximum number of bits to hide data within the shield media without any corruption. Payload is used to measure this ability[6]. Second, keeping the integrity of the shield audio without damage during the hiding process, this could be evaluated by PSNR, BER, and SNR. Third, the security of the stego_technique which may be

achieved by using crypto algorithms, encryption keys, secure algorithms and determining the correct places for hiding.

In this paper, data to be hidden is going to be encoded and compressed using Huffman tree method which also represents characters by binary bits. These binary bits are going to be hidden in transformed audio coefficients. Transforming audio is used to broadcast the effect of hidden data within a wide range of audio samples instead of constraining the effects in small number of selected hearable audio samples. Transformers are used to change the domain of the audio from time to frequency. For any audio, a group of samples in time domain will obtain a coefficient in frequency domain, therefore, modifications made to any coefficient will effect a large spectrum of audio samples in time domain [7].

The suggested technique proposes a new method for hiding encoded, confidential data in determined locations in the transformed covered file. Hiding position is selected using a suitable, predefined private key and the knapsack public key, then a zigzag method is applied on each part of audio samples that is used for hiding and that to ensure more complexity and secrecy for the embedding process.

The paper has many sections, the next section will explain Huffman encoding tree, the third section describes zigzag algorithm. The next section explains knapsack method (the whole cryptosystem). The fifth section explains the suggested technique (determining keys, data encoding, encryption, determining positions, and the process of hiding encrypted data). The sixth section is the implementation of the proposed method. An explanation of the obtained results and an evaluation of the proposed method will be found within the last section.

II. HUFFMAN ENCODING TREE

Huffman code was invented in 1952 to encode characters in binary bits. The method was invented as a lossless compression method which has two major steps: the first step is to find the frequency of each letter in a text. The second step is to encode the characters depending on their frequencies. The method involving a tree structure which is created for characters included in the text. The tree (which is a binary tree) is created by branching in each level to left and right, giving a value of 1 to the right branch and 0 to the left. The creation of the tree depends on one concept and that is: giving the more frequent character a small encoding form while the less frequent character will eventually has a long encoding form. The results of the tree are prefix codes, which means, no two codes starts with the same prefix. Huffman code is used basically in compression but it also could be used in cryptography or even simply for just encoding characters [8].

The first process in the algorithm is to find the frequency of each letter in the text to be encoded then sort the frequencies from the largest to lowest. To build the tree [9]:

1- For each character, add to a priority list and create a leaf node.
2- Since more than one node in the list, do the following:
a- Remove the two lowest frequencies nodes from the list.
b- Create two children nodes and add to the list a node with the summation of the two nodes.
3- The last node remains in the list is the root of the tree and the process of creating a Huffman tree is completed.

Figure 1 shows an example of Huffman tree which is built for the whole English letters depending on the language frequency.

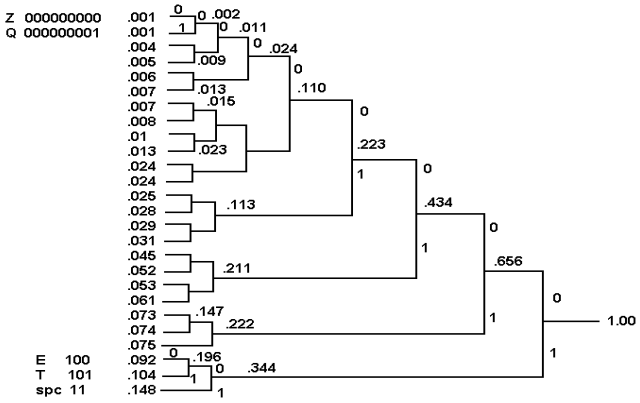


Figure 1: Huffman tree for the English letters

III. ZIGZAG METHOD

Zigzag is a method for reading data in a different confusion way so that the original text is no longer readable or understandable. It is used in encryption and other applications. The characters of a text are arranged in rows like fence then they are read in zigzag sequence [10]. The result text is out of order. Figure 2 shows the sequence of zigzag reading process.

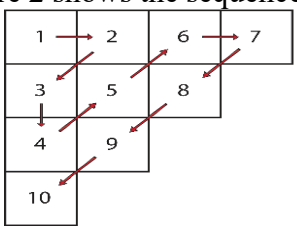


Figure 2: zigzag sequence

IV. KNAPSACK ALGORITHM

Knapsack is a public key algorithm that uses (n) numbers increased in a set.(β). This set is used as a public key. The private key will be (w, q, r) were[11]:

A super increasing sequence will be chosen to encrypt a message of n-bits:

w = (w1, w2,w3, ..., wn)

q and r are integer numbers selected randomly, such that:

$$q > \sum_{i=1}^{n} w_i \quad ,$$

$$\gcd(r,q) = 1$$

n: are nonzero natural numbers.

The selection of q by pervious conditions will obtain the uniqueness of any ciphered text. q shouldn't be any smaller since it may give the same cipher to more than one text. q also should be selected as coprime to r, otherwise no inverse mod q could be obtained. The inverse of r is essential for decryption.

The sequence of the public key will be calculated as:

$$\beta = (\beta 1, \beta 2, \ldots, \beta n)$$

where :

$$\beta_i = r\, w_i \bmod q.$$

The public and the private keys are determined with β and (w, q, r).

A. *Encryption of Knapsack*

A massage of n-bits will be encrypted as:

$$M = \alpha = (\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_n),$$

where each αi represents a bit in the message and

$$\alpha_i \in \{0, 1\}.$$

$$C = \sum_{i=1}^{n} \propto i\, \beta i \mod q\ ,$$

Where: C is the encrypted message.

B. *Decryption of Knapsack*

To obtain the original message from the ciphertext, c‘ will be computed as:

$$c' = r^{-1}\, c \mod q$$

V. THE PROPOSED METHOD

The suggested method has the following steps:

1- Determine keys (Public key for Knapsack algorithm and Private or secrete key).
2- Encode the confidential data using Huffman tree
3- Encrypt the encoded data using Knapsack algorithm.
4- Find the position of hiding (starting position in audio).
5- Transform audio using LWT.
6- Select an audio block with a suitable size.
7- Divide the block into parts each of 25 sample.
8- Convert each part to 5*5 matrix.
9- Determine the order of hiding depending on zigzag method.
10- Encrypted data bits will be hidden within the transformed audio matrices.
11- Return stego_block to its position then apply inverse LWT.

Figure 3 shows the main steps of the proposed method.

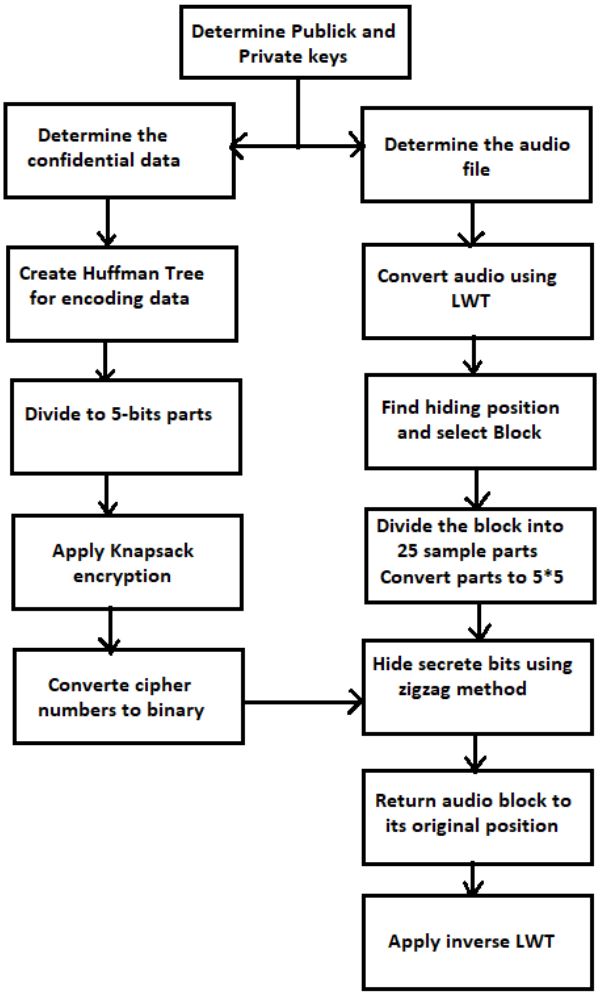


Figure 3: block diagram of the proposed method

A. Key Determination

In this stage both the person who tries to remotely access his (her) data (or application) and other side (bank, application, database,…) agreed on a secrete key to be used in determining the hiding position. Another key is determined for encrypting the data.

In the proposed method the key used in a knapsack algorithm for encryption, and the private predetermined key, will both be used to get the locations where data hidden.

B. Data Encryption

By this step, confidential data will be encrypted during two levels of security as the following:

1- Encoding characters using Huffman tree to obtain compressed, binary notation data.
2- Gathering the binary sequence then divide into parts each of five bits.
3- Knapsack algorithm will be used to encrypt these parts. Then convert coded data to binary to be suitable for the hiding procedure.

C. Selecting positions

Locations (H pos) where data will be hidden is obtained using (the secrete key, the public key, and the audio size) as the following:

Consider S as the audio size, knowing β which is the public key, and the secrete key, K, then:

$$H\ Pos = ((\sum_{i=1}^{n} \beta_i * ((-1)^i+2)) * K) \bmod S$$

…….(2)

Hiding positions are difficult to be exposed since Hpos depends on audio size and a private key which are changed frequently.

The size of confidential data (number of characters) is hidden in the first byte of the selected location, while encrypted data will start from the ninth bit.

D. D. Hiding Process

By this step audio file will transformed to be suitable for hiding process and to reduce the effect on audio after data is hidden. LWT transformer is used and two vectors called approximation and detailed coefficients will be obtained from decomposing audio[6]. The approximation coefficients will be use to hide data bits during hiding process.

Using the selected position, a specific block of audio will be separated for hiding data. The size of this block should equal or greater than the number of bits to be hidden (eight bits for data size plus the number of data bits) and is dividable by 5. The following steps will be applied:

a- Divide the selected audio block into parts each of 25 samples.
b- Convert each part to 5 * 5 matrixes.
c- Hide each bit of data in the LSB of one coefficient using zigzag method. As in figure 4 below:

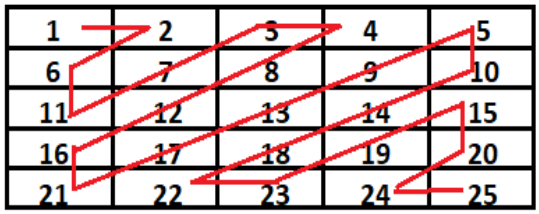


Figure 4: the sequence of hiding using zigzag method

From each coefficient, the least significant bit (LSB) only will be used to hide a data bit, therefore, this hidden data will be neither perceptible nor audible to human perceptual system (HAS).

An inverse transform is applied to audio when the process of hiding is accomplished and that to retrieve the original form of audio.

VI. IMPLEMENTATION OF THE PROPOSED METHOD

Let the confidential data to be hidden within the audio file is:

"Meet me at the park "

Huffman tree will be created for the text by first finding the frequency of each letter:

- E T M A H P R K
  4 3 2 2 1 1 1 1
- E T M A RK H P
  4 3 2 2 2 1 1
- E T M A RK HP
  4 3 2 2 2 2
- E RKHP T M A
  4 4 3 2 2
- E RKHP MA T
  4 4 4 3

- MAT E RKHP
  7 4 4
- ERKHP MAT
  8 7
- ERKHPMAT
  15

The tree will be as in figure 5 below:

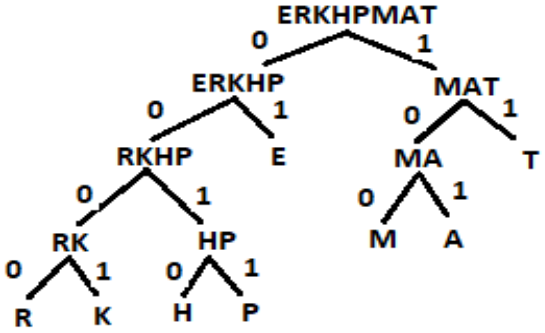


Figure 5: Huffman tree for the text

The sequence of bits results from the tree is shown in table 1 below:

Table 1: results of Huffman tree

Character	Frequency	Code	No. of bits (Frq. * code)
E	4	01	8
T	3	11	6
M	2	100	6
A	2	101	6
H	1	0100	4
R	1	0000	4
K	1	0001	4
P	1	1100	4
Total number of bits =			42

“MEET ME AT THE PARK”

100010111100011011111010001110010100000001

The number of bits obtained from encoding is 42 bits (size of data), in 8-bit binary= 00101010.

These 8-bits will be added to the beginning of the data sequence.

001010101000101111000110111110100011100101000000001

This sequence will be divided into 5-bits parts (length of Knapsack key).

If the sequence bits are not dividable by 5, then add zero(s) to the right of the sequence.

For the knapsack algorithm:

Let:

$$W = \{ 1,2,4,8,16\}$$

Find the summation :

$$p = 1+2+4+8+16 = 31$$

And let:

$$q=34 \ , r=7 \ , \ r^{-1} =5$$

$$\beta_i = = r W_i \bmod q.$$

Then:

$$\beta = \{1*7 \bmod 34 , 2*7 \bmod 34 , 4*7 \bmod 34 , 8*7 \bmod 34 , 16*7 \bmod 34\}$$

We get:

$$\beta = \{ 7,14,28,22,10\}$$

For encryption:

$$M = \alpha = (\alpha_1, \alpha_2, \alpha_3, ..., \alpha_n),$$

Each five bits will be encrypted to produce one number which is converted to binary again to be hidden.

0010101010001011110001101111101000111001010000000 1

00101 , $C = (28+10) \bmod 34 = 4$
01010 , $C=(14+22) \bmod 34 = 2$
00101 , $C=(28+10) \bmod 34 = 4$
11100 , $C=(7+14+28) \bmod 34 = 15$
01101 , $C=(14+28+10) \bmod 34 =18$
11110, $C=(7+14+28+22) \bmod 34 = 3$
10001 , $C=(7+10) \bmod 34 =17$
11001 , $C=(7+14+10) \bmod 34 = 31$
01000 , $C=14 \bmod 34 = 14$
00001 , $C=10 \bmod 34 = 10$

Data = "MEET ME AT THE PARK"
Cipher text =04 02 04 15 18 03 17 31 14 10
To hide the encrypted data within audio, a conversion to binary will be applied.
Audio transformed using LWT. CA and DA are the vectors resulted from the transformation. Only CA will be used to hide bits. The locations of hiding are obtained by:
Let:

$$K = 119 \text{ (secrete key).}$$

If the size of audio = 114412 samples
Then, from equation (2) :

$$Hpos= ((7*1 + 14*3 + 28*1 +22*3 +10*1) * 119) \bmod 114412$$
$$= 18207$$

The bits hiding in CA will start from location (18207) by:

1- Select a block size of 50 sample (the data to be hidden is 50 bit).
2- Divide into two 25-sample parts.

3- Convert each part to 5*5 matrix.
4- Using zigzag method, hide the cipher bits (a bit in one coefficient).

When all the encrypted data bits are hidden, an inverse transform is applied so that audio will return to its original form.

The receiver will get the audio file including hidden data. The receiver is going to apply the same steps as the sender to obtain the cipher bits.

The receiver, at the end of the process, will read the bit from the coefficient instead of writing it (as in sender part). The read bits will be compared with the calculated bits to verify the audio as follows:

For all cipher bits

If

Calculated bit (b) = Extracted(read) bits (d)

Then

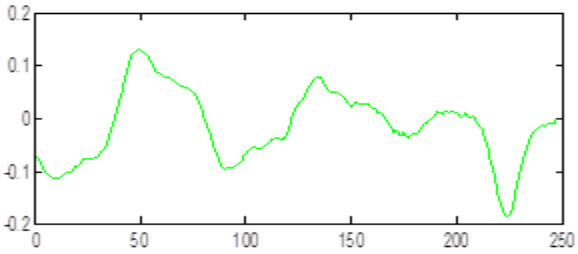
Audio is authorized and remote access is allowed.

Otherwise

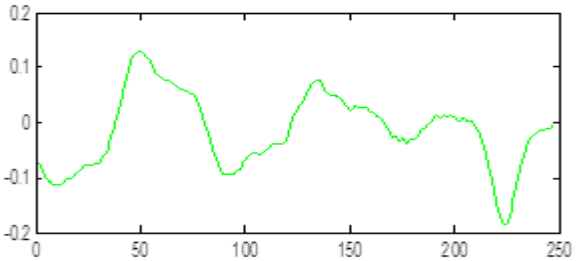
Audio is unauthorized and remote access is denied.

VII. RESULTS AND ANALYSIS

The suggested technique was applied on many audio files, hiding the same confidential data, the frame where the confidential data hidden was not affected as shown by the results. The original frame and the stego frame (256 sample) were the same as shown in figure (6-a and 6-b). This illustrate that hiding data didn't distroy the frame.



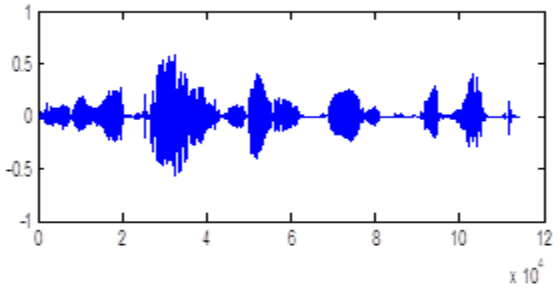
a-Original frame



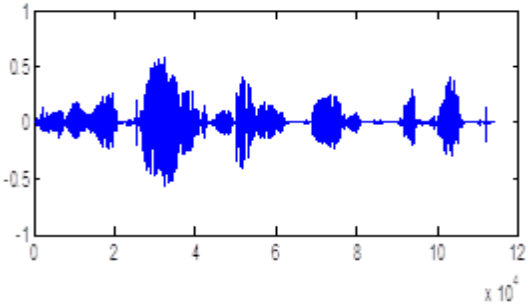
b-Stego_frame

Figure 6: a-Original_Frame, b-Stego_Frame

The stego audio preserve its integrity and no corruption occurred after the hiding of data as illustrated in figure 7.



a-Original_Audio



b-Stego_Audio

Figure 7: a-Original_Audio, b-Stego_Audio

From the suggested method's results we can observe that consistency and integrity of the audio is preserved.

By calculating the evaluation measurements on the steg0 audio, we obtained a very high value for PSNR indicating a small ratio of changes such that no corruption on audio was occurred. The minimum accepted value of SNR is 20dB, the suggested method obtain an SNR value more than that. Table 2 confirms that the results obtained from the suggested method were more than acceptable. The proposed system was applied on five audio files.

Table 2: Results of evaluation metrics

Audio	PSNR	SNR
File1	160.6498	53.5950
File2	177.3341	66.3064
File3	174.1334	61.1064
File4	172.9310	62.4537
File5	172.4377	58.4524

The changing in audio bits could be evaluated using BER. In the suggested method its value was very low and that because the replacement of audio bits with the confidential data bits may cause no effect at all (no change if both audio bit and data bit are either both 1 or both 0).

Distributing the data bits in wide spectrum cause the values of MSR and MAXERR be very low keeping audio quality the same as the original audio. The results are illustrated by table 3 below.

Table 3: Integrity metrics results

Audio	MSE	MAXERR	BER
File1	5.5989 e-012	1.0597 e-004	0.0325
File2	1.2014 e-013	1.0532 e-004	0.0407
File3	2.5104 e-013	1.0345 e-004	0.0650
File4	3.3111 e-013	1.0588 e-004	0.0407
File5	3.7094 e-013	1.0545 e-004	0.0894

The proposed method provides a large payload since the data to be hidden is small comparing to the size of audio files in general (audio duration). Table 4 shows the results of payload for the same five audio files.

Table 4: Results of payload for five audio

Audio	Duration	Payload
File1	00:42	119.04
File2	00:50	155.2
File3	00:44	113.6
File4	00:48	104.1
File5	00:57	185.1

All the locations were data were hidden are chosen depending on (key and audio size) and those factors are changeable frequently. Table 5 shows the position selected in each run.

Table 5: position of hiding

Audio	Audio size	Position
File1	650535	9141
File2	1099640	9772
File3	655010	4417
File4	962808	1935
File5	3303936	3212

Hiding confidential data within a transformed audio kept the audio quality as the original one making the changes inaudible to the audience.

CONCLUSIONS

Audio files (speech, songs, voices,..) are so sensitive by nature to any change. Modifying audio by hiding data is considered as an exigent process and to do so, the quality and integrity of audio should be kept. The proposed technique produced a method for encoding the confidential data using Huffman encoding tree. This method is useful in two ways: encoding and compression, therefore, even if the confidential data is long, it will be reduced because of the Huffman method. The proposed method used knapsack algorithm for encrypting data using a public key. This key, the secrete key and the size of the file were used to find the starting position of hiding. These factors are not constant, either all of them will be changed in each remote accessing, or some of them will be changed. In both cases, the position of hiding will also be changed. Using zigzag method in selecting the order of hiding increases the complexity of the predicting the position or the secrete data. The proposed technique proved to be good by

the results of the measuring factors. BER, MSR and MAXERR were very low while SNR and PSNR were high. The payload was also high which gives the ability to increase the secrete data length if needed.

REFERENCES

[1] Victor Kasacavage, "Complete Book of Remote Access: Connectivity and Security", 1st Edition, Auerbach Publications, 2002.
[2] Qiang Huang, Jazib Frahim, "SSL Remote Access VPNs", Publisher: Cisco Press, 2008.
[3] Wayne Lawso, "Building Cisco Remote Access Networks", 1st Edition, Elsevier,
[4] Mohammed Salem Atom and Osamah Abdulgader Al- Rababah, Alaa Ismat Al-Attili,, "New Technique for Hiding Data in Audio File", IJCSNS International Journal of Computer Science and Network Security, VOL.11 No.4, April 2011.
[5] Raghavan Muthuregunathan, Divya Venkataraman, and Parthiban Rajasekaran, "Cryptanalysis of Knapsack Cipher using Parallel Evolutionary Computing", International Journal of Recent Trends in Engineering, Vol 1, No. 1, May 2009.
[6] Masashi U., Jessada K., Shengbei W., Nhut M., and Ryota M., "Comparative evaluations of inaudible and robust watermarking for digital audio signals", 21st International Congress on Sound and Vibration (ICSV21), Beijing, China, 13-17 July 2014.
[7] S. Murata, Y. Yoshitomi and H.Ishii, "Audio watermarking using Wavelet Transform and Genetic Algorithm for realizing high tolerance to MP3 compression", Journal of Information Security, Vol. 2, pp 99-112, 2011.
[8] Manjeet Kaur, "Lossless Text Data Compression Algorithm Using Modified Huffman Algorithm", International Journal of Advanced Research in Computer Science and Software Engineering, India, July 2015
[9] Rhen Anjerome Bedruz and Ana Riza F. Quiros, "Comparison of Huffman Algorithm and Lempel-Ziv Algorithm for Audio, Image and Text Compression", IEEE International Conference Humanoid, Nanotechnology, Information Technology Communication and Control, Environment and Management (HNICEM). Philippines, 2015.
[10] Mu. Annalakshmi, A. Padmapriya, "Zigzag Ciphers: A Novel Transposition Method", International Journal of Computer Applications, 2013.
[11] , Hans Kellerer, Ulrich Pferschy, David Pisinger ,"Knapsack Problems", Publisher: Springer, 2004.
[12] Joyshree Nath and Asoke Nath, "Advanced Steganography Algorithm using encrypted secrete message", International Journal of Advanced Computer Science and Application (IJACSA) Vol-2 No.3, Page19- 24,March 2011.
[13] R.Kanimozhi and Thanjavur, "A Novel Secure Multimedia Message Hiding Algorithm behind the Image", SSRG International Journal of Computer Science and Engineering (SSRG-IJCSE) – volume1 issue8 October 2014.
[14] Rohit Nagargoje, Manish Raka, Amrut Pol and Prof. B.Gite,, "Embedded Hiding and Extracting Secrete Data in Compress Video File" , International Journal of Embedded Systems, Robotics and Computer Engineering, Volume 2, Number 1, pp. 1-9, 2015.
[15] Sana A. Kadhim, Saad A. Azize, " A proposed method for encrypting and sending confidential data using polynomials", Global Journal of Engineering and Technology Advances, DOI:https://doi.org/ 10.30574/ gjeta.2021.8.3.0122, 2021, 08 (03), 014–019.

[16] Sana Ahmed, Saad Abdualazize. “Embedding Secrete Message in Transformed Voices File using a New Method for Position Selection”, Electronic ISBN: 978-1-5386-9188-5, Print on Demand (PoD) ISBN: 978-1-5386-9189-2. 14 February 2019.
[17] Sana Ahmed, Saad Abdualazize.” Preventing Unauthorized Access to Special Applications using Signed Audio”, International Journal of Civil Engineering and Technology (IJCIET), CiteScore:2.76, SJR:0.246, SNIP:0.153, Impact Factor:9.7820, ISSN:0976-6316.
[18] Sana Ahmed, Saad Abdualazize. A Proposed Method for Image Verification by Encrypted Hidden Text using a Chaotic Polynomial and Random Locations, International Journal of Civil Engineering and Technology (IJCIET), CiteScore:2.76, SJR:0.246, SNIP:0.153, Impact Factor:9.7820, ISSN:0976-6316.