AES Inspired Hex Symbols Steganography for Anti-Forensic Artifacts on Android Devices

Somyia M. Abu Asbeh
Software Engineering Dept.
Princess Sumaya University for Technology
Amman, Jordan

Sarah M. Hammoudeh
Faculty of Medical and Human Sciences
University of Manchester
Manchester, UK

Arab M. Hammoudeh
College of Medicine
University of Sharjah
Sharjah, UAE

***Abstract*—Mobile phones technology has become one of the most common and important technologies that started as a communication tool and then evolved into key reservoirs of personal information and smart applications. With this increased level of complications, increased dangers and increased levels of countermeasures and opposing countermeasures have emerged, such as Mobile Forensics and anti-forensics. One of these anti-forensics tools is steganography, which introduced higher levels of complexity and security against hackers' attacks but simultaneously create obstacles to forensic investigations. In this paper we proposed a new data hiding approach, the AES Inspired Steganography (AIS), which utilizes some AES data encryption concepts while hiding the data using the concept of hex symbols steganography. As the approach is based on the use of multiple encryption steps, the resulting carrier files would be unfathomable without the use of the cipher key agreed upon by the communicating parties. These carrier files can be exchanged amongst android devices and/or computers. Assessments of the proposed approach have proven it to be advantageous over the currently existing steganography approaches in terms of character frequency, security, robustness, length of key, and Compatibility.**

Keywords—Mobile Forensics; Anti-Forensics; Artifact Wiping; Data Hiding; Steganography; AES

I. INTRODUCTION

As mobile phones rapidly evolved from communication means to reservoirs of personal information and smart applications [1], they allowed their users to be exposed to increasing dangers and complexities. Consequently, many fields and technologies have been developed as counter-measures to such dangers. One of these fields is the Mobile Forensics, which aims at collecting and analyzing digital evidence to resolve mobile issues. However, on the other side, opposing measures such as Anti-Forensics technologies have been developed to hinder the use of mobile forensics [2]. One of these anti-forensics tools is steganography.

Steganography systems are utilized to embed secret message in hex symbols, image, audio and video files that can only be discovered by the parties informed of the secret key of the steganography chosen algorithm. Thus, steganography introduces a higher level of complexity that would protect against attacks but at the same time create an obstacle for forensic investigations [3].

This paper will be proposing a new steganography approach inspired by the Advanced Encryption Standard (AES) process, a formal encryption method adopted by the National Institute of Standards and Technology of the US Government, and is accepted worldwide. This encryption method was developed and adopted as a replacement of the Data Encryption Standard (DES) method due to the disadvantages it presented.

The AES encryption method is a 128 bits or 16 bytes block cipher that processes a single block of data at a time and encrypts data through several rounds with the aid of an encryption key. During these ten to fourteen rounds, the data is continuously mixed-up and re-encrypted leading to the increase in the security of the hidden data. A single encryption key is used in the AES method with a length of 128 bits (16 bytes), 192 bits (24 bytes), or 256 bits (32 bytes). The same key would be used for both the encryption and decryption processes which known as symmetric encryption, the opposite of the asymmetric encryption observed in other methods as they utilize two different keys, a public and a private key, in the encryption process [4].

This paper will be introducing some of the currently existing anti-forensics approaches and techniques. Thenceforth, the paper will be presenting the new, AES Inspired Steganography (AIS) approach. This method will be utilizing hex symbols for the embedding of the secret message, similarly to our previously proposed HAS approach [5]. This approach would be applied to purposefully created hex symbol carrier files (using HxD for example) and viewed and edited using the WinHex software.

The AIS approach is proposed to have advantages over the currently existing steganography approaches in its capacity, security and robustness. Capacity refers to the maximum amount of that the stego-medium can contain. Security refers to the ability of the approach and stego-medium to maintain the secrecy of the data by eliminating chances of discovery by third parties. Robustness signifies the ability of the stego-medium to withstand modifications without the loss or compromise of its secretly hidden content [6].

The paper will be presenting background information and related work on anti-forensic techniques, artifact wiping, data hiding, and steganography tools and approaches in sections 2 and 3. Then the paper will be elaborating further on anti-forensics steganography in section 4. The description of the newly proposed AES Inspired Steganography (AIS) is presented in section 5 accompanied by the explanation of the implementation process in section 6. Finally the new approach is analyzed and discussed in section 7.

II. Related Work

Data hiding embeds information in carrier files without changing the general content and format of the file. However, encryption leads to general changes observable by eye to the carrier files reducing the security of the file. Therefore, although encryption increases the difficulty of deciphering the secret messages, the evidence of its existence leaves it prone to attacks. Therefore, combining it with steganography could reduce the vulnerability of this method. AES inspired approached have been developed previously and incorporated into the currently used multimedia steganography methods such as image steganography.

In their paper [7] “A Novel Steganographic Scheme Based on Hash Function Coupled with AES Encryption ” (2014), Rinu et al. presented an AES inspired steganography approach in which the textual data to be hidden is encrypted using the AES approach and embedded in a coloured image using hash based algorithm.

Singh and Attri (2015) proposed another AES inspired steganography approach in their paper [8] “Dual Layer Security of data using LSB Image Steganography Method and AES Encryption Algorithm”. In their work they propose an approach in which data would be embedded in carrier files using LSB image Steganography and encrypted using AES-128 bits encryption resulting in a 2 layered protection of the hidden data. However, this approach has found to result in the invalidation of the stego image used as the carrier file.

Another approach utilizing the concepts of the AES and steganography was presented by Goyal and Sharma in their paper [9] “Proposed AES for Image Steganography in Different Medias” (2014). Their approach utilizes the process of the modified AES (consisting of key expansion, sub bytes modification, shift of rows and mix up of columns) in image-audio steganography. However, the same issue of image invalidation is observed to occur upon the application of this method.

Ramaiya et al. (2013) proposed an image steganography technique based on the AES method in their paper [10] “Secured Steganography Approach Using AES”. The text to be hidden was converted into binary representation in their approach and then embedded into the cover image. The method allows for the use of 128 bit block size of text & 128 bits of Secrete key.

III. Anti-Forensics

Anti-forensics (AF) techniques are used to avoid and eliminate the possibility of evidence detection by the mobile forensics tools [1]. AF techniques and tools are continuously and rapidly evolving. Two major types of Anti-Forensic techniques, artifact wiping and data hiding, will be briefly presented next.

A. Artifact Wiping

Artifact wiping, also known as sanitation, overwrites data files from digital devices permanently erasing them. Some artifact wiping tools, including Binary Code (BC) wipe, Eraser, and Pretty Good Privacy (PGP) wipe, target empty and unallocated spaces [11].

B. Data Hiding

Data hiding tools have been developed to secretly embed and hide undiscoverable data through multiple approaches. These approaches include transferring data to other portable storage devices and then wiping the data from the phone; making data “invisible” and concealing their existence; embedding data in multimedia (hex symbols, image, audio and video) files; and altering file extensions.

IV. The Anti-Forensic Steganography

According to [3], "Steganography is the art and science of hiding information in plain sight". Thus, through steganography, a stego-system unknown to third, uninvolved parties can be created to allow for data exchange under extremely secure conditions. Digitally, data hiding techniques are important tools for the utilization of steganography. Through these tools, hex symbols, image, audio and video steganography can be applied. Steganography techniques are generally categorized into Spatial domain and frequency domain.

A spatial domain technique embeds the information to be concealed in the intensity pixels of the carrier multimedia file. The advantage of this category of techniques is their use of the Least Significant Bit (LSB) algorithms to embed the load of data. However, the drawback is that the majority of the LSB techniques are susceptible to attacks. In frequency domain techniques, on the other hand, images are transformed to frequency components by using some techniques, such as Fast Fourier Transform (FFT), Discrete Cosine Transformation (DCT) or Discrete Wavelet Transform (DWT). Thenceforth, the messages are planted and hidden in some or all of the transformed coefficients [12].

In brief, the process of steganography is commenced through an agreement of two parties on a stego-system and a secret key for the embedding algorithm. The accordingly chosen embedding algorithm would be responsible for allocating the carrier files according to their hexadecimal content. The hexadecimals are modified and replaced with the hexadecimals of the secret message to be exchanged by parties involved. This process prevents any third party lacking the knowledge of the secret key and the chosen embedding algorithm from discovering the embedded data or breaching the carrier file contents [3].

In cryptography, sensitive and secret message is stored and transmitted accorss insecure networks while protected from intruding parties access. Created with a secret key, the encrypted data can only be accessed by the meant parties possessing this key which aids in the deciphering the data [13].

In this paper, we developed an approach that combines concepts from steganography and encryption. The secret message in encrypted using an AES-like process and embedded using the hex symbols algorithm steganography proposed in our previous paper [5]. Subsequent encryption steps are applied to the carrier file as well further to increase the security of the hidden data. We call this approach the AES Inspired Steganography (AIS) (figure 1).

V. The Proposed Aes Ispired Steganography (Ais)

A. AES Inspired Steganography (AIS) Design

In this paper, we will be introducing a new data hiding and encryption method that we call AES Inspired Steganography (AIS). Through this method we aim at overcoming the problem of changing and invalidating the carrier file observed in traditional encryption methods.

In general, this method consists of multiple steps of encryption applied to the secretly hidden message. The message on the other hand is embedded into a hex symbols carrier file, which is divided into embedding matrices and cipher key matrices according to varied patterns chosen by the communicating parties. Furthermore, encryptions and rearrangements are applied to the hidden data before and after being embedded into the carrier file. The use of such variations in hiding and encrypting the data allows for increasing the security measures of the approach.

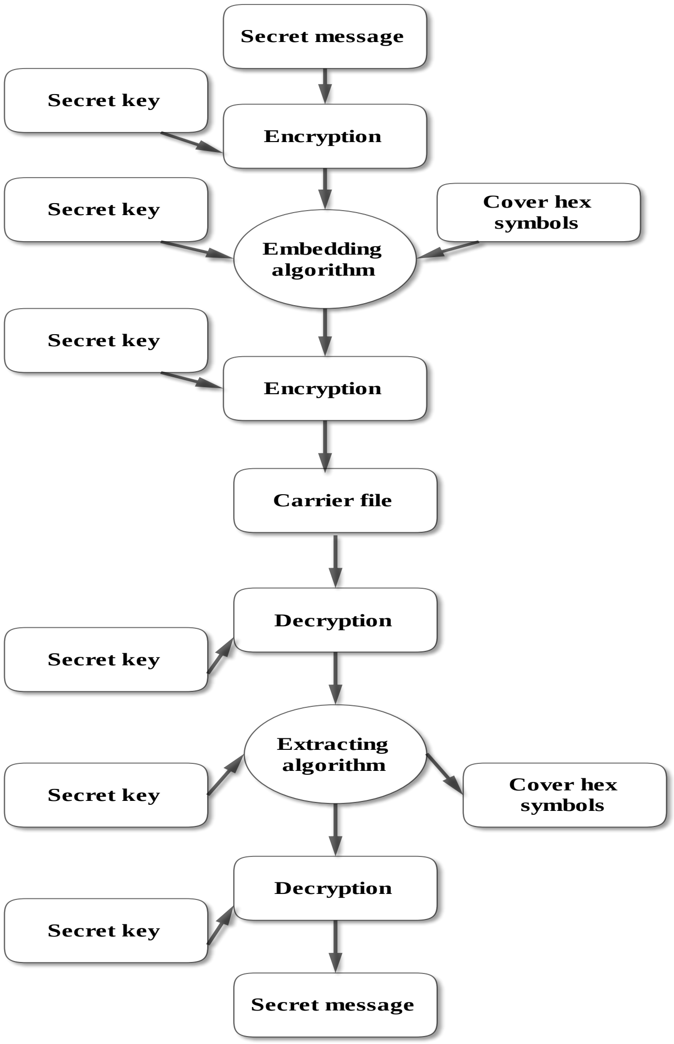


Fig. 1. The general AES Inspired Steganography (AIS) process

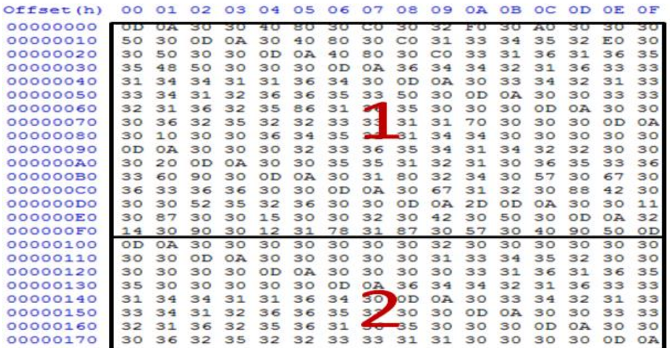
More specifically, the encryption process includes inverting the hexadecimal representation of the of the secret message characters before embedding the message in the carrier file. The embedding process was designed as well to have varying patterns, such that, different combinations of different choices of embedding matrices would be identified to contain the secret message. Furthermore, more specific patterns will be used to identify which characters of the segments would be replaced by the characters of the secret message.

Besides the embedding matrices, cipher key matrices and black segments are included in the matrices divisions causing the decipher process to be even more difficult without a secret key.

After the embedding process, rearrangement of the segments and application of the XOR operation between the cipher key and the embedding matrices further masks the hidden message. These steps are followed by random rearrangements and switching of the rows and columns locations of the matrices. The output version of the carrier file in this case will maintain its validity and integrity. Furthermore, the original content of the carrier file is unfathomable as it mainly consists of random hex symbols. Therefore, the steganography process will not alarm and attract the attention of intruding parties. Additionally, as the file is specifically created for the steganography process, unlike the multimedia files, the possibility of comparisons to originals copy of the file to identify changes is eliminated.

B. AES Inspired Steganography Algorithm (AIS)

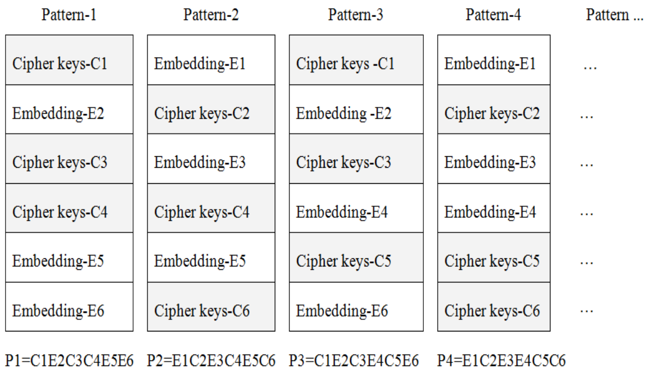
First of all the communicating parties (i.e. sender and receiver) agree upon certain patterns that will be used as keys for embedding and extracting of the secret message content. These patterns are created as follows. A carrier file in the form of symbols is created, for example using HxD software, and converted into hexadecimal using WinHex program. The resulting hex symbols file is segmented into 16x16 matrices and numbered as shown in fig.2.



```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000   0D 0A 30 30 40 80 30 C0 30 32 F0 30 A0 30 30 30
00000010   50 30 0D 0A 30 40 80 30 C0 31 33 34 35 32 E0 30
00000020   30 50 30 30 0D 0A 40 80 30 C0 33 31 36 31 36 35
00000030   35 48 50 30 30 30 0D 0A 36 34 34 32 31 36 33 33
00000040   31 34 34 31 31 36 34 30 0D 0A 30 33 34 32 31 33
00000050   33 34 31 32 36 36 35 33 50 30 0D 0A 30 30 33 33
00000060   32 31 36 32 35 86 31 [illegible] 35 30 30 30 0D 0A 30 30
00000070   30 36 32 35 32 32 33 33 31 31 70 30 30 30 0D 0A
00000080   30 10 30 30 36 34 35 [illegible] 31 34 34 30 30 30 30 30
00000090   0D 0A 30 30 30 32 33 36 35 34 31 34 32 32 30 30
000000A0   30 20 0D 0A 30 30 35 35 31 32 31 30 36 35 33 36
000000B0   33 60 90 30 0D 0A 30 31 80 32 34 30 57 30 67 30
000000C0   36 33 36 36 30 30 0D 0A 30 67 31 32 30 88 42 30
000000D0   30 30 52 35 32 36 30 30 0D 0A 2D 0D 0A 30 30 11
000000E0   30 87 30 30 15 30 30 32 30 42 30 50 30 0D 0A 32
000000F0   14 30 90 30 12 31 78 31 87 30 57 30 40 90 50 0D
00000100   0D 0A 30 30 30 30 30 30 30 32 30 30 30 30 30 30
00000110   30 30 0D 0A 30 30 30 30 30 31 33 34 35 32 30 30
00000120   30 30 30 30 0D 0A 30 30 30 30 33 31 36 31 36 35
00000130   35 30 30 30 30 30 0D 0A 36 34 34 32 31 36 33 33
00000140   31 34 34 31 31 36 34 30 0D 0A 30 33 34 32 31 33
00000150   33 34 31 32 36 36 35 [illegible] 30 30 0D 0A 30 30 33 33
00000160   32 31 36 32 35 36 31 [illegible] 35 30 30 30 0D 0A 30 30
00000170   30 36 32 35 32 32 33 33 31 31 30 30 30 30 0D 0A
```

Fig. 2. The segmented and numbered hex symbols of the carrier file

The segments are then sorted according to the chosen pattern into segments for embedding and segments for the cipher key. Each of these segments are coupled such as each segment used for embedding would be accompanied by a segment for a cipher key. Fig.3 provides an example on the arrangement of the segments in the hex symbols matrices.



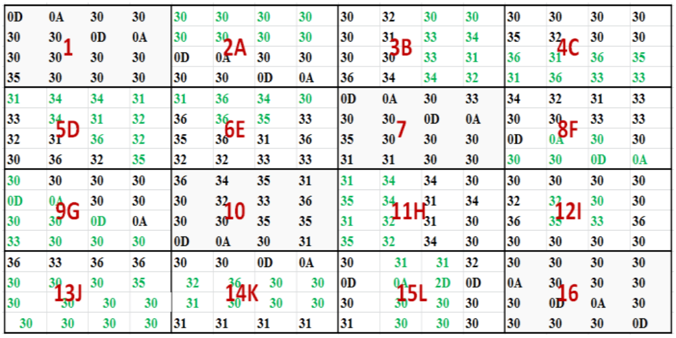
Pattern-1	Pattern-2	Pattern-3	Pattern-4	Pattern ...
Cipher keys-C1	Embedding-E1	Cipher keys -C1	Embedding-E1	...
Embedding-E2	Cipher keys-C2	Embedding -E2	Cipher keys-C2	...
Cipher keys-C3	Embedding-E3	Cipher keys-C3	Embedding-E3	...
Cipher keys-C4	Cipher keys-C4	Embedding-E4	Embedding-E4	...
Embedding-E5	Embedding-E5	Cipher keys-C5	Cipher keys-C5	...
Embedding-E6	Cipher keys-C6	Embedding-E6	Cipher keys-C6	...

P1=C1E2C3C4E5E6 P2=E1C2E3C4E5C6 P3=C1E2C3E4C5E6 P4=E1C2E3E4C5C6

Fig. 3. The division and pattern specification of the embedding and cipher key matrices in the carrier file

From the embedding matrices, specific segments would be chosen to conceal the secret message. Cipher key segments will be as well allocated to each of the chosen embedding segments. The allocation pattern of these chosen segments will be specified in the hiding keys shared between the communicating parties. This paper will be applying pattern 1 (P1) = C1E2C3C4E5E6 as an example. The hex symbol representation of the carrier file is divided into 16×16 matrices which are further divided into16 segments (each forming a 4×4 matrix). These segments are then numbers from 1 to 16.

The segments to be eliminated from the embedding process would be specified as black segments. In our example, matrix E2 includes 4 black segments, 1, 7, 10, and 16. Each of the segments is then to be given a pattern which would be indicated using an alphabet as shown in figure 4.



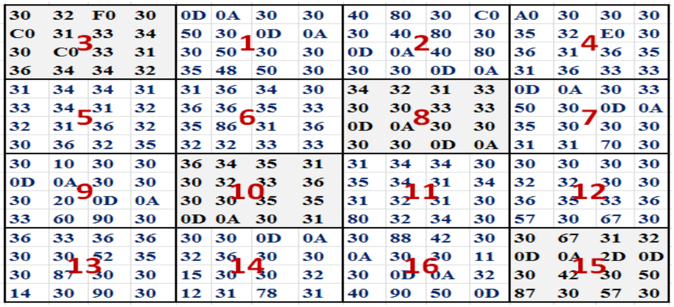
0D	0A	30	30	30	30	30	30	30	32	30	30	30	30	30	30
30	30	0D	0A	30	30	30	30	30	31	33	34	35	32	30	30
30	30	30	30	0D	0A	30	30	30	30	33	31	36	31	36	35
35	30	30	30	30	30	0D	0A	36	34	34	32	31	36	33	33
31	34	34	31	31	36	34	30	0D	0A	30	33	34	32	31	33
33	34	31	32	36	36	35	33	30	30	0D	0A	30	30	33	33
32	31	36	32	35	36	31	36	35	30	30	30	0D	0A	30	30
30	36	32	35	32	32	33	33	31	31	30	30	30	30	0D	0A
30	30	30	30	36	34	35	31	31	34	34	30	30	30	30	30
0D	0A	30	30	30	32	33	36	35	34	31	34	32	32	30	30
30	30	0D	0A	30	30	35	35	31	32	31	30	36	35	33	36
33	30	30	30	0D	0A	30	31	35	32	34	30	30	30	30	30
36	33	36	36	30	30	0D	0A	30	31	31	32	30	30	30	30
30	30	30	35	32	36	30	30	0D	0A	2D	0D	0A	30	30	30
30	30	30	30	31	30	30	30	30	30	30	30	30	0D	0A	30
30	30	30	30	31	31	31	31	31	30	30	30	30	30	30	0D

Fig. 4. A demontration of a chosen pattern (pattern 1_E2) applied to an embedding metrix of the carrier file. The figure indicates the chosen black segments indicated with only numbers while other segments are indicated by a combination of numbers and alphabets

The input secret message is then converted into a hexadecimal representation; therefore each character of its content will be in the form of a two digits hex symbol. These digits representing each character of the secret message are then inverted. For example if the letter 'n' was to be hidden, it will be first converted to the hex code number 63, then inverted to become 36. The resulting inverted hex representations will then be embedded into matrices segments of the carrier file that will have a unique pattern as shown in Fig.4.

Similarly the cipher key matrices will be divided to 16 segments each formed of a 4x4 matrix, which will be numbered from 1 to 16 (figures 9 and 10). Black segments will be chosen as well, in this case, 3, 8, 10, and 15.

The black segments in the cipher key matrix will be relocated to match the locations of those in the embedding matrix to which the cipher key matrix is coupled. For example segment number 3 will be moved forward and relocated prior to segment number 1. Segments number 1 and 2 will then be shifted 1 block foreword each and so on (fig.5).



30	32	F0	30	0D	0A	30	30	40	80	30	C0	A0	30	30	30
C0	31	33	34	50	30	0D	0A	30	40	80	30	35	32	E0	30
30	C0	33	31	30	50	30	30	0D	0A	40	80	36	31	36	35
36	34	34	32	35	48	50	30	30	30	0D	0A	31	36	33	33
31	34	34	31	31	36	34	30	34	32	31	33	0D	0A	30	33
33	34	31	32	36	36	35	33	30	30	33	33	50	30	0D	0A
32	31	36	32	35	86	31	36	0D	0A	30	30	35	30	30	30
30	36	32	35	32	32	33	33	30	30	0D	0A	31	31	70	30
30	10	30	30	36	34	35	31	31	34	34	30	30	30	30	30
0D	0A	30	30	30	32	33	36	35	34	31	34	32	32	30	30
30	20	0D	0A	30	30	35	35	31	32	31	30	36	35	33	36
33	60	90	30	0D	0A	30	31	80	32	34	30	57	30	67	30
36	33	36	36	30	30	0D	0A	30	88	42	30	30	67	31	32
30	30	52	35	32	36	30	30	0A	30	30	11	0D	0A	2D	0D
30	87	30	30	15	30	30	32	30	0D	0A	32	30	42	30	50
14	30	90	30	12	31	78	31	40	90	50	0D	87	30	57	30

Fig. 5. Illustration of the re-arrangement of the cipher key matrix to allow the black segments to have superimposing location as thise of the embedding metrix

The hex decimals are then converted to binary representation both the cipher key and the embedding matrices, specifically the decimals representing the secret message content in the embedding segments and those opposing them in the cipher key segments. After the conversion to the binary representations, the decimals are processed using the XOR operation and the resulting binary representation is then converted back to hexadecimal representation. The process is illustrated in table I.

TABLE I. THE XOR OPERATION

Input (hexadecimal)	0A	XOR	47
Convert to binary	00001010	X	01000111
Output(binary)	01001101		
Convert to hexadecimal	4d		

The 16 segments are then randomly rearranged and the order would be described in a table and allocated a certain code such as Random(S) E2 =3,4,1,2,7,8,5,6,11,12,9,10,15,16,13,14.

The contents of these matrices are relocated by exchanging rows with columns in order to increase the difficulty for hackers.

Similarly, the segments of the cipher key matrices would be rearranged randomly such as Random(S) C1=1,3,4,2,8,5,6,7,11,9,12,10,16,13,15,14. The contents of these matrices are relocated by exchanging rows and columns. Then the hexadecimal representation of each of the characters would be inverted.

The hex symbols book is a table like reference shared between the communicating parties with the key information to decipher the encrypted message. This book would include the arrangement of the embedding and cipher key matrices (E# / C#) for a collection of chosen pattern. Furthermore each pattern would be accompanied with information about the black segments, the original allocations of each of the segments before the randomization processes and the patterns (indicated by the alphabet representations) used in each of the segments to indicate the characters containing the concealed hex decimals as shown in table II.

TABLE II. EXAMPLE OF THE SHARED KEY HEX SYMBOL CODEBOOK

Symbols	S	O	M
Pattern - number	P1=C1E2C3C4E5E6	….	…

Embedding segment block number	E2-B = 1-7-10-16
	E5-B =3-8-10-15
	E6-B =2-5-11-13
		….	…

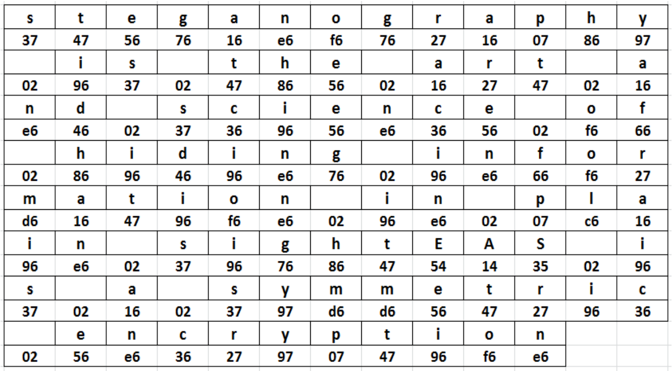
Cipher key segment block number	C1-B=3-8-10-15
	C3-B=2-5-11-13
	C4-B=1-7-15-1
		….	…

Pattern allocated to 4x4 segments	E2-W-1 =ABCDEFGHIJKL
	E5-W-2 =JKLGHIDEFABC
	E6-W-3 =DGJAEKBHLCFI
		….	…

Random key	C1=1,3,4,2,8,5,6,7,11, 9,12,10,16,13,15,14 . .	……	… …
	E2=3,4,1,2,7,8,5,6,11, 12,9,10,15,16,13,14 . .	…… .	… … …

VI. IMPLEMENTATION

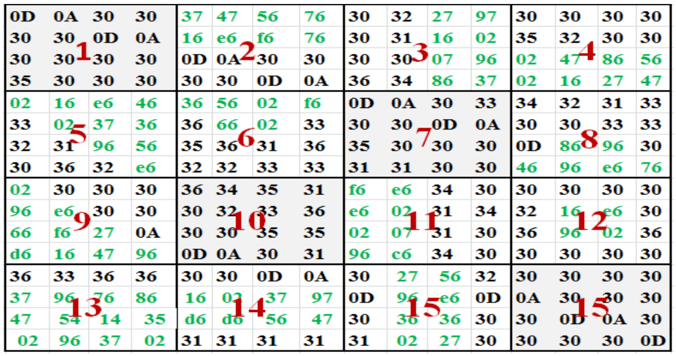
An example of the proposed AIS (AES Inspired Steganography) Approach will be presented in this section in which the secret message "Steganography is the art and science of hiding information in plain sight. EAS is a symmetric encryption" will be embedded in a hex symbols carrier file. The characters of the message are converted to into the hexadecimal representation to begin with. Each letter of the message would be represented by two hexadecimal character components. The two hexadecimals forming each character are then inverted as shown in fig.6, for eample 73 would become 37.



s	t	e	g	a	n	o	g	r	a	p	h	y
37	47	56	76	16	e6	f6	76	27	16	07	86	97
	i	s		t	h	e		a	r	t		a
02	96	37	02	47	86	56	02	16	27	47	02	16
n	d		s	c	i	e	n	c	e		o	f
e6	46	02	37	36	96	56	e6	36	56	02	f6	66
	h	i	d	i	n	g		i	n	f	o	r
02	86	96	46	96	e6	76	02	96	e6	66	f6	27
m	a	t	i	o	n		i	n		p	l	a
d6	16	47	96	f6	e6	02	96	e6	02	07	c6	16
i	n		s	i	g	h	t	E	A	S		i
96	e6	02	37	96	76	86	47	54	14	35	02	96
s		a		s	y	m	m	e	t	r	i	c
37	02	16	02	37	97	d6	d6	56	47	27	96	36
	e	n	c	r	y	p	t	i	o	n		
02	56	e6	36	27	97	07	47	96	f6	e6		

Fig. 6. Secret message hex symbols after inversion

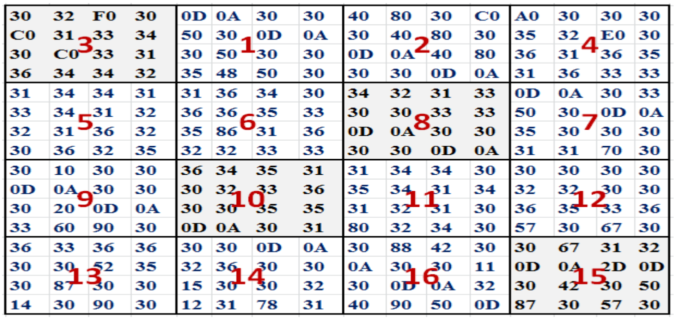
The resulting inverted, hexadecimal representation of secret message's characters is embedded into embedding segment 2 (E2) according to our choice of 'S', which represents pattern-1 (Fig.7). As explained before, the secret message will be embedded after the choice of the black segments and the embedding pattern of the secret message in each of the segments of the embedding matrix (shown in green).



0D	0A	30	30	37	47	56	76	30	32	27	97	30	30	30	30
30	30	0D	0A	16	e6	f6	76	30	31	16	02	35	32	30	30
30	30	30	30	0D	0A	30	30	30	30	07	96	02	47	86	56
35	30	30	30	30	30	0D	0A	36	34	86	37	02	16	27	47
02	16	e6	46	36	56	02	f6	0D	0A	30	33	34	32	31	33
33	02	37	36	36	66	02	33	30	30	0D	0A	30	30	33	33
32	31	96	56	35	36	31	36	35	30	30	30	0D	86	96	30
30	36	32	e6	32	32	33	33	31	31	30	30	46	96	e6	76
02	30	30	30	36	34	35	31	f6	e6	34	30	30	30	30	30
96	e6	30	30	30	32	33	36	e6	02	31	34	32	16	e6	30
66	f6	27	0A	30	30	35	35	02	07	31	30	36	96	02	36
d6	16	47	96	0D	0A	30	31	96	c6	34	30	30	30	30	30
36	33	36	36	30	30	0D	0A	30	27	56	32	30	30	30	30
37	96	76	86	16	02	37	97	0D	96	e6	0D	0A	30	30	30
47	54	14	35	d6	d6	56	47	30	36	36	30	30	0D	0A	30
02	96	37	02	31	31	31	31	31	02	27	30	30	30	30	0D

Fig. 7. Secret message hex symbols after inversion. Illustration of the embedding matrix (E2) after the embedding of the secret message according to 'S' (pattern-1). the embedded secret message is represented by bold green characters while the black segments are shaded with light grey

Similarly, the cipher key matrices are prepared by choosing the black segments and rearranging the segments of the matrix (figure 8). Accordingly, the locations of the black segments in both the embedding and cipher key matrices would be superimposing as explained earlier.



30	32	F0	30	0D	0A	30	30	40	80	30	C0	A0	30	30	30
C0	31	33	34	50	30	0D	0A	30	40	80	30	35	32	E0	30
30	C0	33	31	30	50	30	30	0D	0A	40	80	36	31	36	35
36	34	34	32	35	48	50	30	30	30	0D	0A	31	36	33	33
31	34	34	31	31	36	34	30	34	32	31	33	0D	0A	30	33
33	34	31	32	36	36	35	33	30	30	33	33	50	30	0D	0A
32	31	36	32	35	86	31	36	0D	0A	30	30	35	30	30	30
30	36	32	35	32	32	33	33	30	30	0D	0A	31	31	70	30
30	10	30	30	36	34	35	31	31	34	34	30	30	30	30	30
0D	0A	30	30	30	32	33	36	35	34	31	34	32	32	30	30
30	20	0D	0A	30	30	35	35	31	32	31	30	36	35	33	36
33	60	90	30	0D	0A	30	31	80	32	34	30	57	30	67	30
36	33	36	36	30	30	0D	0A	30	88	42	30	30	67	31	32
30	30	52	35	32	36	30	30	0A	30	30	11	0D	0A	2D	0D
30	87	30	30	15	30	30	32	30	0D	0A	32	30	42	30	50
14	30	90	30	12	31	78	31	40	90	50	0D	87	30	57	30

Fig. 8. Rearrangement of the cipher key matrix in order for the black segments (shaded with light grey) to have superimposing location with those of the embedding matrix

Subsequently, the XOR operation is applied to the matrices as shown in figure 9 to produce the newly encrypted form of the embedding matrices (figure 10).

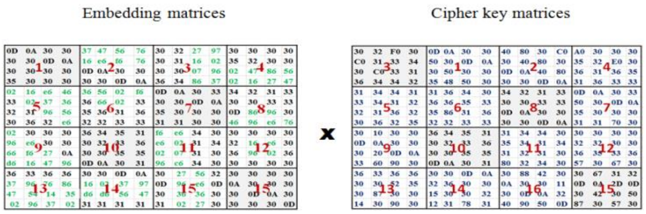
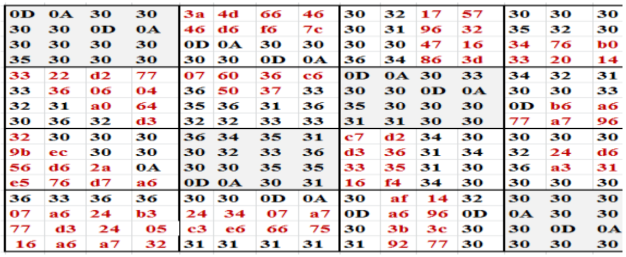


Fig. 9. The application of the XOR operation to the secret message containing characters of the embedding matrix and their opposing characters on the cipher key metrix

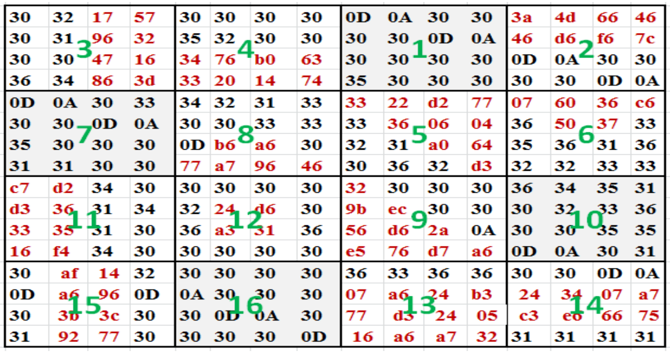


0D	0A	30	30	3a	4d	66	46	30	32	17	57	30	30	30
30	30	0D	0A	46	d6	f6	7c	30	31	96	32	35	32	30
30	30	30	30	0D	0A	30	30	30	30	47	16	34	76	b0
35	30	30	30	30	30	0D	0A	36	34	86	3d	33	20	14
33	22	d2	77	07	60	36	c6	0D	0A	30	33	34	32	31
33	36	06	04	36	50	37	33	30	30	0D	0A	30	30	33
32	31	a0	64	35	36	31	36	35	30	30	30	0D	b6	a6
30	36	32	d3	32	32	33	33	31	31	30	30	77	a7	96
32	30	30	30	36	34	35	31	c7	d2	34	30	30	30	30
9b	ec	30	30	30	32	33	36	d3	36	31	34	32	24	d6
56	d6	2a	0A	30	30	35	35	33	35	31	30	36	a3	31
e5	76	d7	a6	0D	0A	30	31	16	f4	34	30	30	30	30
36	33	36	36	30	30	0D	0A	30	af	14	32	30	30	30
07	a6	24	b3	24	34	07	a7	0D	a6	96	0D	0A	30	30
77	d3	24	05	c3	e6	66	75	30	3b	3c	30	30	0D	0A
16	a6	a7	32	31	31	31	31	31	92	77	30	30	30	30

Fig. 10.

Fig. 11. The resulting embedding matrix from the XOR operation

The segments of the new embedding matrix are rearranged randomly (figure 11).


3
4
1
2
7
8
5
6
11
12
9
10
15
16
13
14

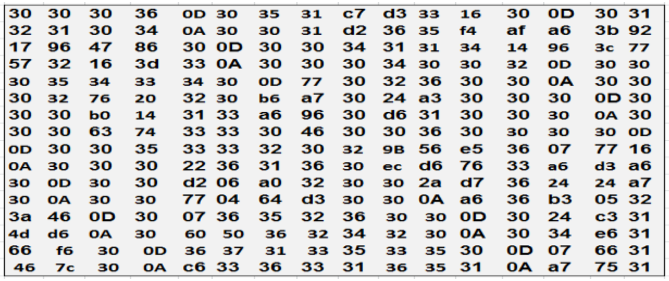
30	32	17	57	30	30	30	30	0D	0A	30	30	3a	4d	66	46
30	31	96	32	35	32	30	30	30	30	0D	0A	46	d6	f6	7c
30	30	47	16	34	76	b0	63	30	30	30	30	0D	0A	30	30
36	34	86	3d	33	20	14	74	35	30	30	30	30	30	0D	0A
0D	0A	30	33	34	32	31	33	33	22	d2	77	07	60	36	c6
30	30	0D	0A	30	30	33	33	33	36	06	04	36	50	37	33
35	30	30	30	0D	b6	a6	30	32	31	a0	64	35	36	31	36
31	31	30	30	77	a7	96	46	30	36	32	d3	32	32	33	33
c7	d2	34	30	30	30	30	30	32	30	30	30	36	34	35	31
d3	36	31	34	32	24	d6	30	9b	ec	30	30	30	32	33	36
33	35	31	30	36	a3	31	36	56	d6	2a	0A	30	30	35	35
16	f4	34	30	30	30	30	30	e5	76	d7	a6	0D	0A	30	31
30	af	14	32	30	30	30	30	36	33	36	36	30	30	0D	0A
0D	a6	96	0D	0A	30	30	30	07	a6	24	b3	24	34	07	a7
30	3b	3c	30	30	0D	0A	30	77	d3	24	05	c3	e6	66	75
31	92	77	30	30	30	30	0D	16	a6	a7	32	31	31	31	31

Fig. 12. Random rearrangement of the embedding matrix

Finally, one more rearrangement is applied to the embedding matrix by interchanging the positions of the rows and columns of the embedding matrix as shown in Fig.12. This is achieved by flipping the whole segment elements around the diagonal as shown in equation 1:

$$x'_{ij} = x_{ji} \qquad (1)$$

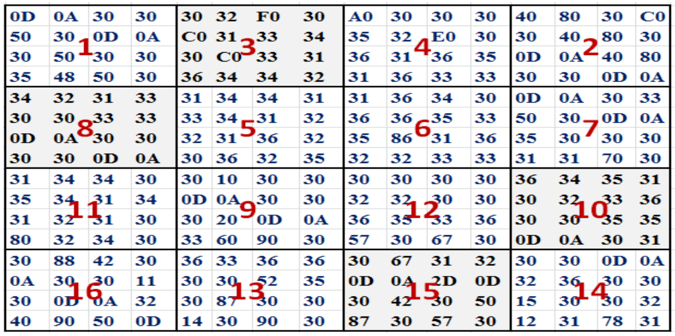
Where x'_{ij} are the new matrix elements of the stego-file and x_{ji} are the old matrix elements.



30	30	30	36	0D	30	35	31	c7	d3	33	16	30	0D	30	31
32	31	30	34	0A	30	30	31	d2	36	35	f4	af	a6	3b	92
17	96	47	86	30	0D	30	30	34	31	31	34	14	96	3c	77
57	32	16	3d	33	0A	30	30	30	34	30	30	32	0D	30	30
30	35	34	33	34	30	0D	77	30	32	36	30	30	0A	30	30
30	32	76	20	32	30	b6	a7	30	24	a3	30	30	30	0D	30
30	30	b0	14	31	33	a6	96	30	d6	31	30	30	30	0A	30
30	30	63	74	33	33	30	46	30	30	36	30	30	30	30	0D
0D	30	30	35	33	33	32	30	32	9B	56	e5	36	07	77	16
0A	30	30	30	22	36	31	36	30	ec	d6	76	33	a6	d3	a6
30	0D	30	30	d2	06	a0	32	30	30	2a	d7	36	24	24	a7
30	0A	30	30	77	04	64	d3	30	30	0A	a6	36	b3	05	32
3a	46	0D	30	07	36	35	32	36	30	30	0D	30	24	c3	31
4d	d6	0A	30	60	50	36	32	34	32	30	0A	30	34	e6	31
66	f6	30	0D	36	37	31	33	35	33	35	30	0D	07	66	31
46	7c	30	0A	c6	33	36	33	31	36	35	31	0A	a7	75	31

Fig. 13. The final form of the embedding segment after the exhange of the locations of the rows and columns

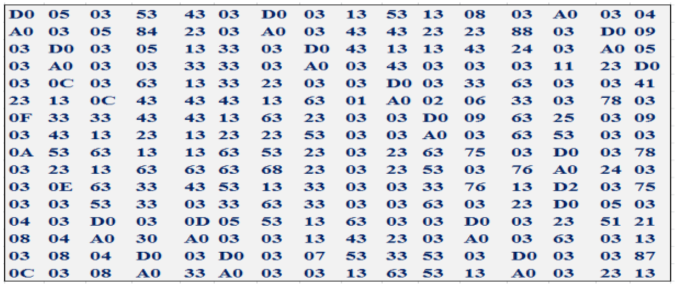
Rearrangements are applied to the cipher key matrices as well. Initially, a random arrangement is applied (figure 13), followed by interchanging the locations of the rows and columns.


1
3
4
2
8
5
6
7
11
9
12
10
16
13
15
14

0D	0A	30	30	30	32	F0	30	A0	30	30	30	40	80	30	C0
50	30	0D	0A	C0	31	33	34	35	32	E0	30	30	40	80	30
30	50	30	30	30	C0	33	31	36	31	36	35	0D	0A	40	80
35	48	50	30	36	34	34	32	31	36	33	33	30	30	0D	0A
34	32	31	33	31	34	34	31	31	36	34	30	0D	0A	30	33
30	30	33	33	33	34	31	32	36	36	35	33	50	30	0D	0A
0D	0A	30	30	32	31	36	32	35	86	31	36	35	30	30	30
30	30	0D	0A	30	36	32	35	32	32	33	33	31	31	70	30
31	34	34	30	30	10	30	30	30	30	30	30	36	34	35	31
35	34	31	34	0D	0A	30	30	32	32	30	30	30	32	33	36
31	32	31	30	30	20	0D	0A	36	35	33	36	30	30	35	35
80	32	34	30	33	60	90	30	57	30	67	30	0D	0A	30	31
30	88	42	30	36	33	36	36	30	67	31	32	30	30	0D	0A
0A	30	30	11	30	30	52	35	0D	0A	2D	0D	32	36	30	30
30	0D	0A	32	30	87	30	30	30	42	30	50	15	30	30	32
40	90	50	0D	14	30	90	30	87	30	57	30	12	31	78	31

Fig. 14. Random rearrangement of the cipher key matrix

Finally the hexadecimal representation of each of the characters in the cipher key matrix is inverted (figure 14).



D0	05	03	53	43	03	D0	03	13	53	13	08	03	A0	03	04
A0	03	05	84	23	03	A0	03	43	43	23	23	88	03	D0	09
03	D0	03	05	13	33	03	D0	43	13	13	43	24	03	A0	05
03	A0	03	03	33	33	03	A0	03	43	03	03	03	11	23	D0
03	0C	03	63	13	33	23	03	03	D0	03	33	63	03	03	41
23	13	0C	43	43	43	13	63	01	A0	02	06	33	03	78	03
0F	33	33	43	43	13	63	23	03	03	D0	09	63	25	03	09
03	43	13	23	13	23	23	53	03	03	A0	03	63	53	03	03
0A	53	63	13	13	63	53	23	03	23	63	75	03	D0	03	78
03	23	13	63	63	63	68	23	03	23	53	03	76	A0	24	03
03	0E	63	33	43	53	13	33	03	03	33	76	13	D2	03	75
03	03	53	33	03	33	63	33	03	03	63	03	23	D0	05	03
04	03	D0	03	0D	05	53	13	63	03	03	D0	03	23	51	21
08	04	A0	30	A0	03	03	13	43	23	03	A0	03	63	03	13
03	08	04	D0	03	D0	03	07	53	33	53	03	D0	03	03	87
0C	03	08	A0	33	A0	03	03	13	63	53	13	A0	03	23	13

Fig. 15. The inversion of the hexadecimal representaions of the cipher key matrix characters

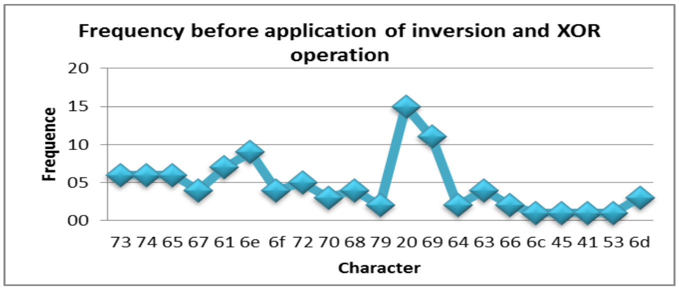
With these final rearrangements the carrier files would be ready for the safe exchange between the communicating parties. With the aid of the secret keys and codebooks shared by the communicating parties, the steps would be retraced to recover the key message.

VII. Analysis and Discussion

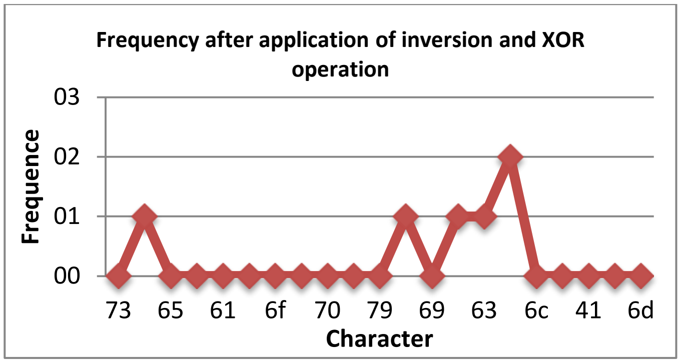
The frequency, degree of safety, robustness, key length, compatibility and capacity of the proposed AIS scheme were analyzed as follows.

A. *Frequency*

As the two hexadecimal character components of the hex symbol are inverted and processes according to the XOR operation with the cipher key, the calculated frequency of occurrence before and after the application of these changes will differ (fig. 15-a and 1-b).



(a)



(b)

Fig. 16. Character frequency assessment of the embedded message: (a) before and (b) after the inversion of the characters and application of the XOR operation with the superimposing characters of the cipher key matrix

From the compiled frequencies analysis in table III and fig. 16, clear differences were observed in the frequencies of the characters before and after the application of the changes. This observation positively indictes the high level of security against third attacking parties, which is expected to be even further increased with the increase in the length of the secret message.

TABLE III. CHARACTER FREQUENCY BEOFRE (F.B) AND AFTER (F.A) INVERSION AND APPLICATION OF THE XOR WITH THEIR COUPLED CIPHER KEY SEGMENTS

Character	F.B	F.A
73	06	00
74	06	01
65	06	00
67	04	00
61	07	00
6e	09	00
6f	04	00
72	05	00
70	03	00
68	04	00
79	02	00
20	15	01
69	11	00
64	02	01
63	04	01
66	02	02
6c	01	00
45	01	00
41	01	00
53	01	00
6d	03	00

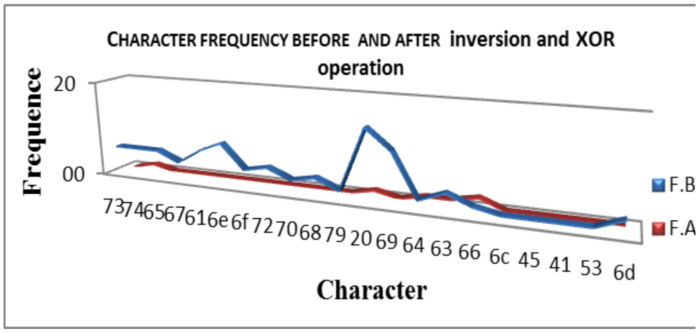
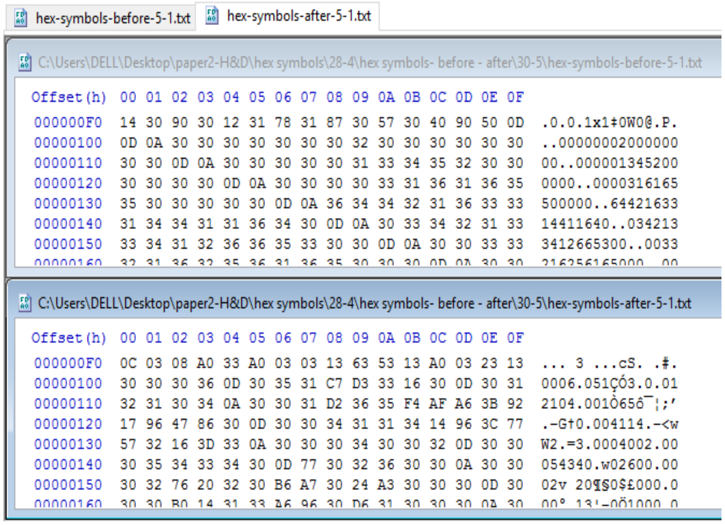


Fig. 17. Character frequency assessment of the embedded message: (a) before (F.B) and (b) after (F.A) the inversion of the characters and application of the XOR operation with the superimposing characters of the cipher key matrix

B. Using WinHex

The use of WinHex to formulate the hex symbols during the hiding process is advantageous as the content will be difficult to trace and compare with previous versions. This advantage becomes more critical as frequent rearrangements of the hex symbols are applied throughout the steganograohy procedure.

A comparison was conducted between the hex symbols in the carrier file (viewed using WinHex) prior to and after the encryption of the secret message. As shown fig.17, a complete change occurred in the hex symbols content of the carrier file, making it impossible for a third party to detect any traces of the message.



C:\Users\DELL\Desktop\paper2-H&D\hex symbols\28-4\hex symbols- before - after\30-5\hex-symbols-before-5-1.txt

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

000000F0  14 30 90 30 12 31 78 31 87 30 57 30 40 90 50 0D  .0.0.1x1‡0W0@.P.
00000100  0D 0A 30 30 30 30 30 30 30 32 30 30 30 30 30 30  ..00000002000000
00000110  30 30 0D 0A 30 30 30 30 30 31 33 34 35 32 30 30  00..000001345200
00000120  30 30 30 30 0D 0A 30 30 30 30 33 31 36 31 36 35  0000..0000316165
00000130  35 30 30 30 30 30 0D 0A 36 34 34 32 31 36 33 33  500000..64421633
00000140  31 34 34 31 31 36 34 30 0D 0A 30 33 34 32 31 33  14411640..034213
00000150  33 34 31 32 36 36 35 33 30 30 0D 0A 30 30 33 33  3412665300..0033
00000160  32 31 36 32 35 36 31 36 35 30 30 30 0D 0A 30 30  216256165000..00
```

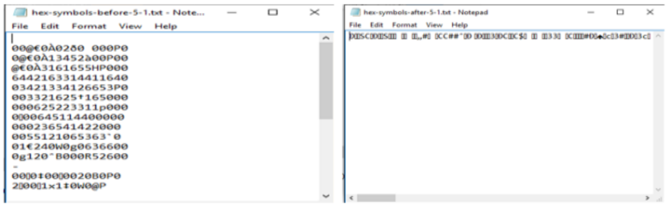
C:\Users\DELL\Desktop\paper2-H&D\hex symbols\28-4\hex symbols- before - after\30-5\hex-symbols-after-5-1.txt

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

000000F0  0C 03 08 A0 33 A0 03 03 13 63 53 13 A0 03 23 13  ... 3 ...cS. .#.
00000100  30 30 30 36 0D 30 35 31 C7 D3 33 16 30 0D 30 31  0006.051ÇÓ3.0.01
00000110  32 31 30 34 0A 30 30 31 D2 36 35 F4 AF A6 3B 92  2104.001Ò65ô¯¦;’
00000120  17 96 47 86 30 0D 30 30 34 31 31 34 14 96 3C 77  .–G†0.004114.–<w
00000130  57 32 16 3D 33 0A 30 30 30 34 30 30 32 0D 30 30  W2.=3.0004002.00
00000140  30 35 34 33 34 30 0D 77 30 32 36 30 30 0A 30 30  054340.w02600.00
00000150  30 32 76 20 32 30 B6 A7 30 24 A3 30 30 30 0D 30  02v 20¶§0$£000.0
00000160  30 30 B0 14 31 33 A6 96 30 D6 31 30 30 30 0A 30  00°.13¦–0Ö1000.0
```

Fig. 18. Comparison between the Hex Symbols before and after the encryption of the secret message

The carrier file content was additionally compared without being viewed using WinHex (Figure 18). The comparison has shown the maintenance of the integrity of the file before and after the encryption process. Such as, the encryption would not cause the invalidity of the file after the encryption as seen in other approaches [8][9].


hex-symbols-before-5-1.txt - Note...
File
Edit
Format
View
Help
hex-symbols-after-5-1.txt - Notepad
File
Edit
Format
View
Help

```
00@€0À0280 000P0
0@€0À13452à00P00
@€0À3161655HP000
6442163314411640
03421334126653P0
003321625†165000
00062522331 1p000
0▯00645114400000
000236541422000
005512106536З`0
01€240W0g0636600
0g120ˆB000R52600
-
00▯0‡00▯0020B0P0
2▯00▯1x1‡0W0@P
```

```
[illegible]
```

(a)

(b)

Fig. 19. Hex symbols content of the carrier file (a) before and (b) after the inversion of the characters and application of the XOR operation with the superimposing characters of the cipher key matrix

Moreover the alteration of the hex symbols by the inversion of each character element doesn't increase the original file size, leaving it stable and unchanging in terms of elements number. Furthermore, the use of random numbers to select the segments provides an extra complication against deciphering the hidden text. A comparison between available steganalysis tools and hex symbols is presented in Table IV.

TABLE IV. COMPARISON BETWEEN STEGANALYSIS AND HEX SYMBOLS

STEGANALYSIS TOOLS		HEX SYMBOLES
OurSecret OmniHide BDV DataHider Max file encryption Masker StegoStick	These tools work by embedding information within videos by attaching it bluntly to the end of the file EOF [3].	Hex symbols substitutes the hexadecimal precisely on the same position.
OurSecret	This signature can be found after the last byte of the authentic unmodified file.	A valid signature similar to OurSecret does not appear.
OmniHide Pro	White space characters tailing the initial sequence of bytes.	Hex symbols do not show the name of the embedded file.

C. The safety and security against encryption traces

The proposed approach does not require the use of any external encryption and data hiding tools, but rather utilizes tools embedded in usually commonly software such as Microsoft Excel. Therefore, traces of steganography and encryption specified tools on the communicating parties' personal devices would be hard to detect. Therefore, the elimination of such a potent traces source would reduce the risk of alarming the attacking parties. Moreover, the hex symbol file extension can be changed to mislead hackers and investigators.

D. Compression

When tested under compression options such as WinRAR and ZIP file formats, the carrier file has been found to resist changes in size and content. This resistance indicates the robustness of the proposed approach against modification that could be applied to the file which steadily maintaining the file's integrity and content safety.

E. Key length

As shown in table V, in our example, the size of the hex symbols was 1536 bytes and the size of the secret message used was 336 bytes while the size of the cipher key was equally 336 bytes as suggested earlier with regards to the pattern size required for embedding the secret message. Any increase in the required pattern size for embedding the secret message will result in a simultaneous and equal increase in the size of the secret message and cipher key. Therefore, in our approach we have been able to include all the methods of ciphering texts; the transposition (permutation), the substitution and the one-time pad. Achieving the one-time pad is a very significant strength of our new approach, as it was developed to have an equal size for both the cipher key and the secret message regardless of the size of the secret message. Furthermore, the longer the cipher key and the message are, the harder it is to identify the cipher key by intruding parties. Moreover, the use of the hex symbols carrier file and hexadecimal representation of its content allows for a higher embedding capacity in comparison with the use of binary representation.

TABLE V. LENGTHS OF THE EMBEDDING AND CIPHER KEY SEGMENTS, THE HEX SYMBOLS IN THE CARRIER FILE, THE SECRET MESSAGE AND THE CIPHER KEY

Segment for embedding		**embedding**		**Segment for cipher key**		**Cipher key**	
name	Size/ byte	byte	bit	name	Size/ byte	byte	bit
E1	256	96	768	C1	256	96	768
E2	256	120	960	C2	256	120	960
E3	256	120	960	C3	256	120	960
	768	336	2688		768	336	2688

Hex symbols		**embedding**		**Cipher key**	
byte	bit	byte	bit	byte	bit
1536	12288	336	2688	336	2688

F. Compatibility & Capacity

The use of Hex symbols was very compatible with the use of the AES concept as both approaches are based on the use of the hexadecimal representation of the content or target text. Such compatibility allows for coherence and flexibility as well as high storage capacity due to the use of the hexadecimal representation in comparison with the methods utilizing binary representations. The use of the hexadecimal representation has the advantage of the higher robustness in comparison to the binary representation as the writing and modification of the hexadecimal representation is relatively easier. Furthermore, basing the approach on the hexadecimal representation reduced the length of the code needed in comparison to the binary representation.

In this proposed approach we have added several steps to increase the complexity degree of the hidden message. First, the approach chooses certain segments and hexadecimal characters to embed in while leaving some without embedded character. Therefore, upon the continuous rearrangement of the matrices, the location of the secret message characters would be hard to identify. Second, we added the idea of using random black segments to increase the complexity of the encryption. Finally, we inverted the hexadecimal representations of the characters at several occasions. The combination of these modifications with the multiple steps of encryption and the steganography process resulted in a very complex system that would only be deciphered through the use of the secret key book.

VIII. CONCLUSION & FUTURE WORK

The AES Inspired Steganography (AIS) approach that we propose in this paper represents a modified, improved version of both the AES and the steganography approaches, as it overcomes the weaknesses of each of the techniques through the strength of the other. The approach utilizes the multi-step encryption idea of the AES in combination with the safe data hiding concept of the steganography to conceal secret messages in hex symbols carrier files. This approach has been proven to have advantages over the currently existing steganography approaches in terms of capacity, safety and robustness.

In the future, this approach can be developed further to increase its complexity. The length of the secret message as well as the cipher key could as well be further modified and increased to increase the capacity of the approach. Furthermore, the approach could be developed to be incorporated into other applications and techniques.

Acknowledgment

We thank prof. Nihad Yusuf – Dean of King Abdullah I Faculty of Graduate Studies and Scientific Research, prof. Arafat Awajan – Dean of King Hussein Faculty for Computing Sciences, and Dr. Saqer Abdel Rahim – Head of the computer science department in King Hussein Faculty for computing Sciences, for their continuous support of this work.

References

[1] A. Distefano, G. Me and F. Pace, "Android anti-forensics through a local paradigm," Digital Investigation, vol. 7, pp. S83-S94, August 2010.

[2] K. Dahbur and B. Mohammad "The Anti-Forensics Challenge," Proceedings of the 2011 International Conference on Intelligent Semantic Web-Services and Applications - ISWSA '11, ACM Press, April 2011.

[3] T. Sloan and J. Hernandez-Castro, "Forensic analysis of video steganography tools," PeerJ Computer Science, vol. 1, pp. e7, May 2015.

[4] "Introduction to AES Encryption,"2016, Townsend Security. [online]. available: https://townsendsecurity.com/sites/default/files/AES_Introduction.pdf

[5] S. M. Abu Asbeh, H. A. Al-Sewadi, S. M. Hammoudeh, A. M. Hammoudeh, "Hex Symbols Algorithm for Anti-Forensic Artifacts On Android Devices," International Journal of Advanced Computer Science and Applications (IJACSA), vol. 7, no. 4, April 2016.

[6] S. Sirsikar and A. Deshpande, "Steganographic Tools for BMP Image Format," International Journal of Computer Science & Emerging Technologies (IJCSET), vol. 2, pp. 200-204, February 2011.

[7] Manoj gowtham.G.V, Senthur.T, Sivasankaran.M, Vikram.M,Bharatha Sreeja.G, "AES BASED STEGANOGRAPHY," International Journal of Application or Innovation in Engineering & Management (IJAIEM), Volume 2, Issue 1, January 2013.

[8] Satwinder Singh and Varinder Kaur Attri, "Dual Layer Security of data using LSB Image Steganography Method and AES Encryption Algorithm," International Journal of Signal Processing, Image Processing and Pattern Recognition, Vol. 8, No. 5 (2015).

[9] Yojna Goyal , Manmohan Sharma, "PROPOSED AES FOR IMAGE STEGANOGRAPHY IN DIFFERENT MEDIAS," IJRET: International Journal of Research in Engineering and Technology, Volume: 03 Issue: 10 | Oct-2014.

[10] MANOJ RAMAIYA, NAVEEN HEMRAJANI and ANIL KISHORE SAXENA, "SECURED STEGANOGRAPHY APPROACH USING AES," International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR), Vol. 3, Issue 3, Aug 2013.

[11] P. A. Kotsopoulos and Y. C. Stamatiou, "Uncovering Mobile Phone Users' Malicious Activities Using Open Source Tools," Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on, Istanbul, pp. 927-933, August 2012.

[12] K.Dasgupta1, J.K. Mandal and P.Dutta "HASH BASED LEAST SIGNIFICANT BIT TECHNIQUE FOR VIDEO STEGANOGRAPHY(HLSB) " International Journal of Security, Privacy and Trust Management (IJSPTM), Vol. 1, No 2, April 2012.

[13] Obaida Mohammad Awad Al-Hazaimeh, "A NEW APPROACH FOR COMPLEX ENCRYPTING AND DECRYPTING DATA," International Journal of Computer Networks & Communications (IJCNC) Vol.5, No.2, March 2013.