IMAGE PROCESSING IN SECURE MANNER USING VLSI

Ms.R.VANMATHI, Ms.M.PRABAVATHI (UG Students) and Mr. S.JOSEPH (Asst. Prof)
C.R. ENGINEERING COLLEGE,
Madurai, Tamilnadu
India

***Abstract*- Advanced Encryption Standard (AES) have been widely used in data encryption and decryption. The application of steganography in encrypted data is reported in a recent article. Any type of data can be practically hidden inside any image, without detectably affecting the quality of the image. This can be done by substituting some information of the image with the secret information in carefully chosen ways. The receiver can get the secret data by applying reverse transform. The AES algorithm is a symmetric block cipher that can encrypt (encipher) and decrypt (decipher) information. Encryption converts data to an unintelligible form called cipher text; decrypting the cipher text converts the data back into its original form, called plaintext. For practical use, VLSI architectures of the proposed algorithms are designed and realized using Xilinx ISE VLSI software for hardware implementation.**

Index Terms- image encryption, image decryption, VLSl, Steganography, Information hiding.

I. Introduction

This standard specifies the Rijndael algorithm [3] and [4], a symmetric block cipher that can process data blocks of 128 bits, using cipher keys with lengths of 128, 192, and 256 bits Rijndael was designed to handle additional block sizes and key lengths, and however they are not adopted in this standard. Throughout the remainder of this standard, the algorithm specified herein will be referred to as "the AES algorithm." The algorithm may be used with the three different key lengths indicated above, and therefore these different "flavors" may be referred to as "AES-128", "AES-192", and"AES-256" The input and output for the AES algorithm each consist of sequences of 128 bits (digits with values of 0 or 1). These sequences will sometimes be referred to as blocks and the number of bits they contain will be referred to as their length. Cipher Key for the AES algorithm is a sequence of 128, 192 or 256 bits. Other input, output and Cipher Key lengths are not permitted by this standard. The bits within such sequences will be numbered starting at zero and ending at one less than the sequence length (block length or key length). The number I attached to a bit is known as its index and will be in one of the ranges 0 □□ i < 128, 0 □
depending on the block length and key length. Due to the increase in computer power, the internet and with the development of digital signal processing, information theory and coding theory, steganography became digital. Steganography has created an atmosphere of corporate vigilance that has generated various interesting applications, as a result its continuing evolution is guaranteed. Instead a chip is designed that automatically embeds the given data in an image. If the user has a flash drive, the chip receives the data, embeds it within an image sand sends It back to the flash drive. This is the encryption module. A retrieving module is also designed that retrieves the embedded data from the cover. This paper aims at explaining one such method of secret sharing through images and developing a hardware prototype for the process.

II. Algorithm Specification

For the AES algorithm, the length of the input block, the output block and the State is 128 bits. This is represented by Nb= 4, which reflects the number of 32-bit words (number of columns) in the State. For the AES algorithm, the length of the Cipher Key, K, is 128, 192, or 256 bits. The key length is represented by Nk= 4, 6, or 8, Which reflects the number of 32 bit words (number of columns) in the Cipher Key. For the AES algorithm, the number of rounds to be performed during the execution of the algorithm is dependent on the key size. The number of rounds is represented by Nr, where Nr =10 when Nk= 4, Nr = 12 when Nk= 6, and Nr = 14 when Nk= 8. The only Key-Block- combinations that conform to this standard are given. For implementation issues relating to the key length, block size and number of rounds.

III. CIPHER

The Cipher is described in the pseudo code in T individual transformations Sub Bytes(), shift Rows(),Mix Columns() and Add Round Key() – process the State and are described in the following sub sections. N_r rounds are identical with the exception of the final round, which does not include the Mix Columns() transformation. Appendix B presents an example of the Cipher, showing values for the State array at the Beginning of each round and after the application of the four transformations described in the following sections.

```
KeyExpansion(byte key[4*Nk], word w[Nb*(Nr+1)],Nk)

Cipher(byte in[4*Nb],byte out[4*Nb],word w[Nb*(Nr+1)])
begin
     byte state[4,Nb]
     state = in
     AddRoundKey(state, w[0, Nb-1])
     for round = 1 step 1 to Nr-1
          SubBytes(state)
          ShiftRows(state)
          MixColumns(state)
          AddRoundKey(state,w[round*Nb,(round+1)*Nb-1])
     end for
     SubBytes(state)
     ShiftRows(state)
     AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
     out = state
end
```

IV. RIJNDAEL ALGORITHM

Rijndael algorithm is an iterated block cipher [9] supporting a variable data block and a variable key length of 128, 192 or 256 bits. The algorithm consists of three distinct phases: (i) an AES algorithm, the number of rounds to be performed during the execution of the algorithm is dependent on the key size. The number of rounds is represented by Nr, where Nr =10 when Nk= 4, Nr = 12 when Nk= 6, and Nr = 14 when Nk= 8. The only Key-Block- Round combinations that conform to this standard are given. For implementation issues relating to the key length, block size and number of rounds.

	Key Length *(Nk words)*	Block Size *(Nb words)*	Number of Rounds *(Nr)*
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

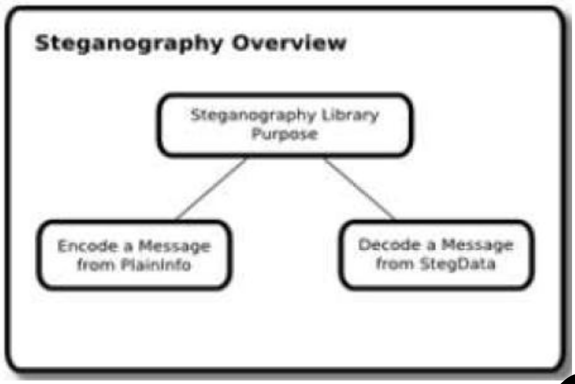
(i) Initial data/key addition, (ii) nine (128-bits) eleven (192-bits) or thirteen (256-bits) standard rounds, (iii) a final round which is a variation of a standard round. The number of standard rounds depends on the data block and key length. If the maximum length of the data blocker key is 128, 192 or 256, then the number of rounds is 10, 12 or 14, respectively. The initial key is expanded to generate the round keys, each of size equal to block length. Each round of the algorithm receives a new round key from the key schedule B. module. Each standard round includes four fundamental algebraic function transformations on arrays of bytes. These transformations are: byte substitution, shift row, mix column, and round key addition. The final round of the algorithm is similar to the standard round, except that it does not have Mix Column operation. Decryption is performed by the application of the inverse transformations of the round functions. The sequence of operations for the standard round function differs from encryption. The computational performance differs between encryption and decryption because the inverse transformations in the round function is more complex than the corresponding transformation for encryption.

V. THE STEGANOGRAPHY

Now we have the encrypted data that is to be sent over the channel to the receiver such that itis not hampered. So the cipher text obtained in the above step is taken and hidden into an image using the process of Steganography. Steganography serves two main purposes:
5.1 Encode message of plain info.

5.2. Decode Message from steg Data

Plain info is an input this is normally derived from encrypted data. Cover data is another input. In our case, it is a picture message. The minimum amount of Cover data is related to the amount of plain info. The process one is of encoding the plain info with the cover data. The cover data will normally have to be Prepared in some way to encode the data.


Steganography Overview
Steganography Library Purpose
Encode a Message from PlainInfo
Decode a Message from StegData

The output is the resulting steg data. The Input is: Plain Info (encrypted data from DES) Image (cover data) an image file is merely a binary file containing a binary representation of the color or light intensity of each picture element (pixel) comprising the image. Images typically use either 8-bit or 24-bit color. When using 8-bit color, there is a definition of up to 256 colors forming a palette for this image, each color denoted by an 8-bit value. A 24-bit color scheme, as the term suggests, uses 24 bits per pixel and provides a much better set of colors. In this case, each pixel is represented by three bytes, each byte representing the intensity of the three primary colors red, green, and blue (RGB), respectively. The size of an image file, then, is directly related to the number of pixels and the granularity of the color definition. A typical 640x480 pix image using a palette of 256 colors would require a file about 307 KB in size (640 • 480 bytes),whereas a 1024x768 pix high resolution 24-bit color image would result in a 2.36 MB file (1024 • 768 • 3 bytes). GIF and 8-bit BMP files employ what is known as lossless compression, a scheme that allows the software to exactly reconstruct the original image. JPEG, on the other hand, uses lossy compression, which means that the expanded image is very nearly the same as the original but not an exact duplicate. While both methods allow computers to save storage space, lossless compression is much better suited to applications where the integrity of the original information must be maintained, such as Steganography. While JPEG can be used for stego applications, it is more common to embed data in GIF or BMP files. The Process is: We have to check cover Data amount against plain Info. Encode plain Info with cover Data (picture). The approach to hiding data within an image file used here is called least significant bit (LSB) insertion. In this method, we can take the Binary representation of the hidden data and overwrite the LSB of each byte within the cover image. If we are using 24-bit color, the amount of change will be minimal and indiscernible to the human eye. As an example, suppose that we have three adjacent pixels (nine bytes) with the following RGB encoding:

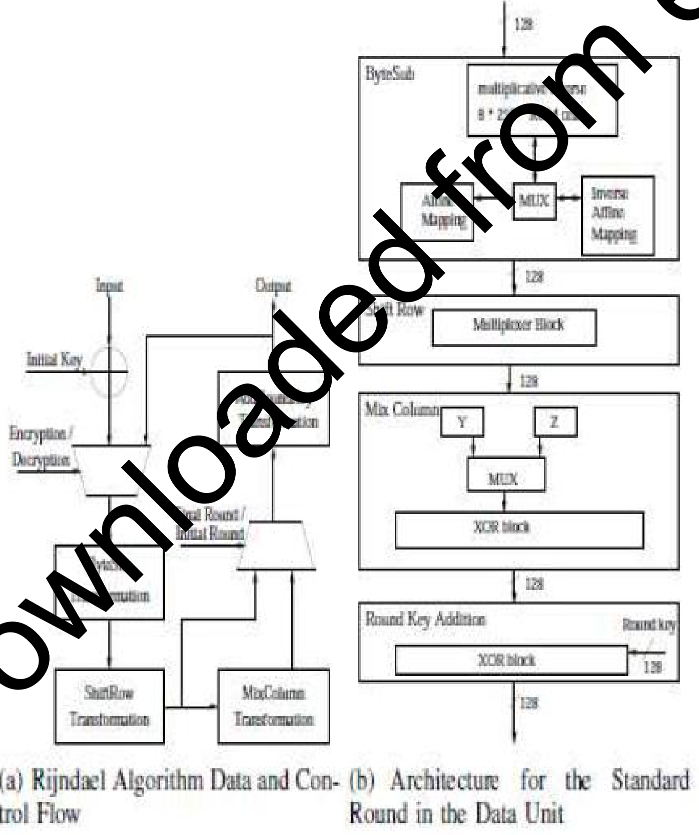
10010101 00001101 11001001
10010110 00001111 11001010
10011111 00010000 11001011

Hereby the Steganography process is consummated with the hiding of the data in the image. Now this image can be sent to the receiver over an unsecure channel.

VI. THE PROPOSED

VLSIARCHITECTURE FOR RIJNDAEL

The proposed architecture showing the order of operation and control between the transformation. A. Architecture of the Data Unit. The data unit consists of: the initial round of key Nr ¡ 1 standard rounds, and a final round .The architecture for a standard round composed of four basic blocks is shown. For each block, both the transformation and the inverse transformation needed for encryption and decryption, respectively are performed using the same hardware resources. This implementation generates one set of sub key and reuses it for calculating all other sub keys in real-time.1) Byte Sub: In this architecture each block is replaced by its substitution in an S Box table consisting of the multiplicative inverse of each byte of the block state in the finite field GF(28).In order to overcome the performancebottleneck.2) Shift Row: In this transformation the rows of the block state are shifted over different offsets. The amount of shifts is determined by the block length. The proposed architecture implements the shift row operation using combinational logic considering the offset by which a row should be shifted.3) Mix Column: In this transformation each column of the block state is considered as a polynomial over GF(28). It is multiplied with a constant polynomial C(x) or D(x) over a finite field in encryption or decryption, respectively. In hardware, the multiplication by the corresponding polynomial is done by XOR operations and multiplication of a block by X. This is implemented using a multiplexer; the control being the MSB is 1 or 0. The equations implemented in hardware for Mix Column in encryption and decryption are as follows.4) AddRoundKey: In this transformation the round key obtained from the key scheduler is XORed with the block state obtained from the Mix Column transformation or Shift Row transformation based on the type of round being implemented. In the standard round, the round key is XORed with the output obtained from the MixColumn transformation. In the final round the round key is XORed



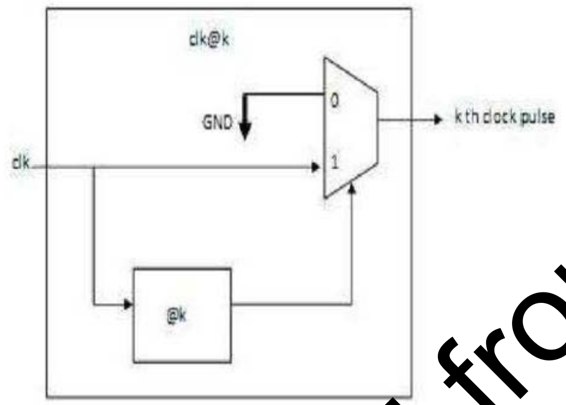
(a) Rijndael Algorithm Data and Control Flow (b) Architecture for the Standard Round in the Data Unit

Fig. 1. Top Level View of the Rijndael

with the output obtained from the Shift Row transformation. In the initial round, bitwise XOR operation is performed between the initial round key and the initial state.

VI.1.EMBEDDER HARDWARE ARCHITECTURE

The embedding module embeds the bits of text data in the least significant bit of the cover image data and gives the stego image with the text embedded in it. This has an architecture which utilizes D-Flip Flops, Multiplexers, etc. as. The two D Flip Flops take image data and text data as inputs. The text input is taken from the memory which has the text data. The image read frequency is eight times the text read frequency. The blocks labeled _@k„ where k=5,6,7,8 give out a logical _1„ from 5th,6th,7thand 8th clock pulses till the end of the 8thclockpulse in a 8-period cycle of image read frequency. This output signal selects the text bits to be output at the time interval corresponding to the least _n„ significant bits. The value of k is chosen according to the relation k=9-n.The @k blocks are made of 8bit shift registers which are initially loaded with all _1„s in the first (9-k) bit positions and zeros in all other bit positions. The output of the LSB is fed back to the MSB of the shift register. This arrangement makes sure that a logical _1„Is given out from start of kth clock pulse to end of 8th clock pulse. The 4:1 multiplexer selects the output of any one of the @k blocks based on the input n_minus_1 which is a binary representation of (n-1) where text has to be embedded in n lsbs of the image data. The 2:1 multiplexer selects either image data bit or the text data bit depending on the output of the 4:1 multiplexer. Thus, n bit LSB embedding is possible using this simple architecture. The above block diagram, the block labeled—@k‖gives out clock pulses at bit intervals of range [k,8].



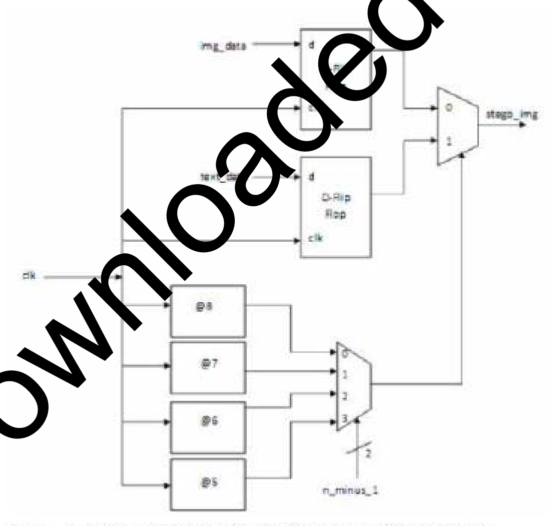


Fig.1. A block diagram showing hardware architecture of the embedder module.

Here, k<9 and k € N. _n„ is the number of least significant bits to be used for embedding secret information. This hardware expects the cover image to be a two dimensional sampled image, with 8 bit resolution for each pixel.

VI.2.RECOVERER HARDWAREARCHITECTURE

In the recoverer block diagram as shown , the block labeled ―clk@k‖gives out the clock pulse at bit interval k. Here, k<9 and k€N. _n„ is the number of least significant bits to be used for embedding secret information. This hardware outputs the secret information when a stego image is input in it.

The recovery module recovers the data embedded in the least significant bit of the stego image data and gives the text data back in its original form. This also has an architecture which utilizes D-Flip Flops, Multiplexers, etc. as shown in Fig-2. The D-Flip Flop takes the stego image data as the input. This D Flip Flop is triggered by a clock only at the time intervals corresponding to n LSBs so that the text data is continuously given out at the output of D-Flip

Flop. The blocks labeled _clk@k„ where k=5,6,7,8 give out a replica of the input clock pulses from kth to 8th clock pulse in a 8-period cycle of image read frequency. The output signal selects the text bits to be output at the time interval corresponding to the _n„ least significant bits. The value of k is same as in the case of embedding. The clk@k blocks have the architecture as shown in Fig-3. They are made of corresponding _@k„ blocks along with a

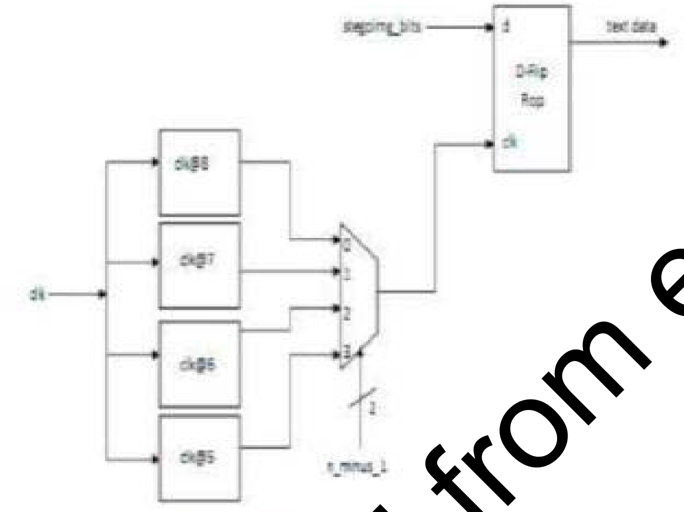


Fig.2. A block diagram showing hardware architecture of the recoverer

2:1multiplexer. This multiplexer selects either a logical _0„ or the input clock based on the output from _@k„ block. This arrangement makes sure that clock pulses are given only from kth to 8th bit interval of the image data. The 4:1 multiplexer selects the output of any one of the clk@k blocks based on the input n_minus_1 which is a binary representation for (n-1) where text has to be embedded in n lsbs of the image data. The output of this 4:1 multiplexer is given to the clk input of the D-Flip Flop which outputs the text portion of the input stego image. Thus,this architecture enables recovery.

VII.STEGANOGRAPHY DISTORTIONANALYSIS

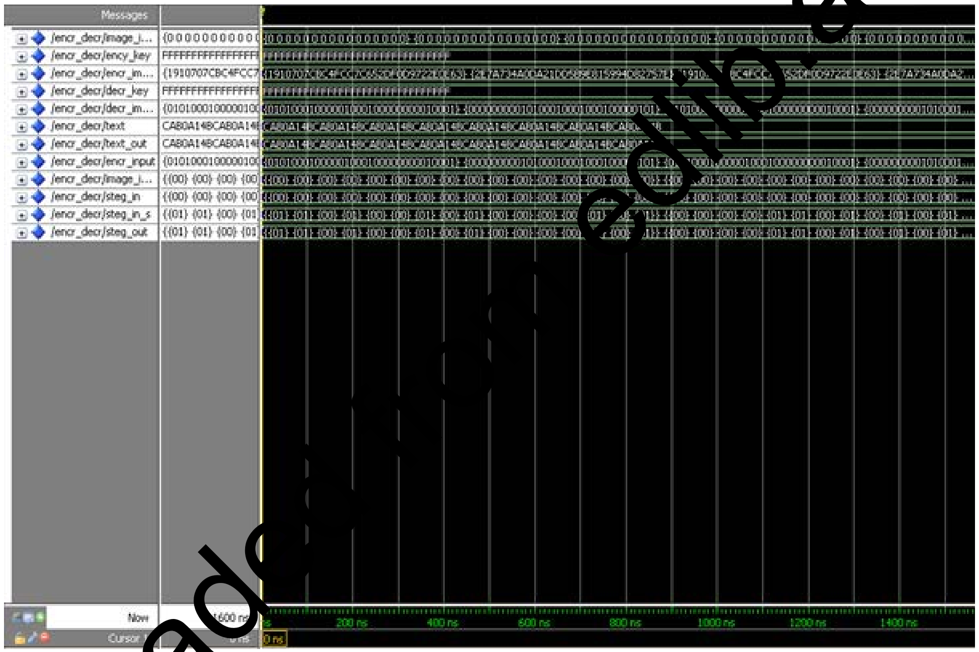
Distortion is measured by means of two parameters namely, Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR). The MSE is calculated by using the equation, where M and N denote the total number of pixels in the horizontal and the vertical dimensions of the image Xi, j represents the pixels in the original image and Yi, j, represents the pixels of the stego-image. The Peak Signal to Noise Ratio(PSNR) is expressed as The PSNR is calculated using the equation where Imax is the intensity value of each pixel which is equal to 255 for 8 bit gray scale images. Higher the value of PSNR better the image quality Distortion Analysis of stego image using the software based secret sharing algorithm with 100% embedding of data gave the following results.

$$MSE = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(X_{i,j} - Y_{i,j}\right)^{2} \qquad (1)$$

$$PSNR = 10\log_{10}\left(\frac{I_{\max}^{2}}{MSE}\right)dB \qquad (2)$$

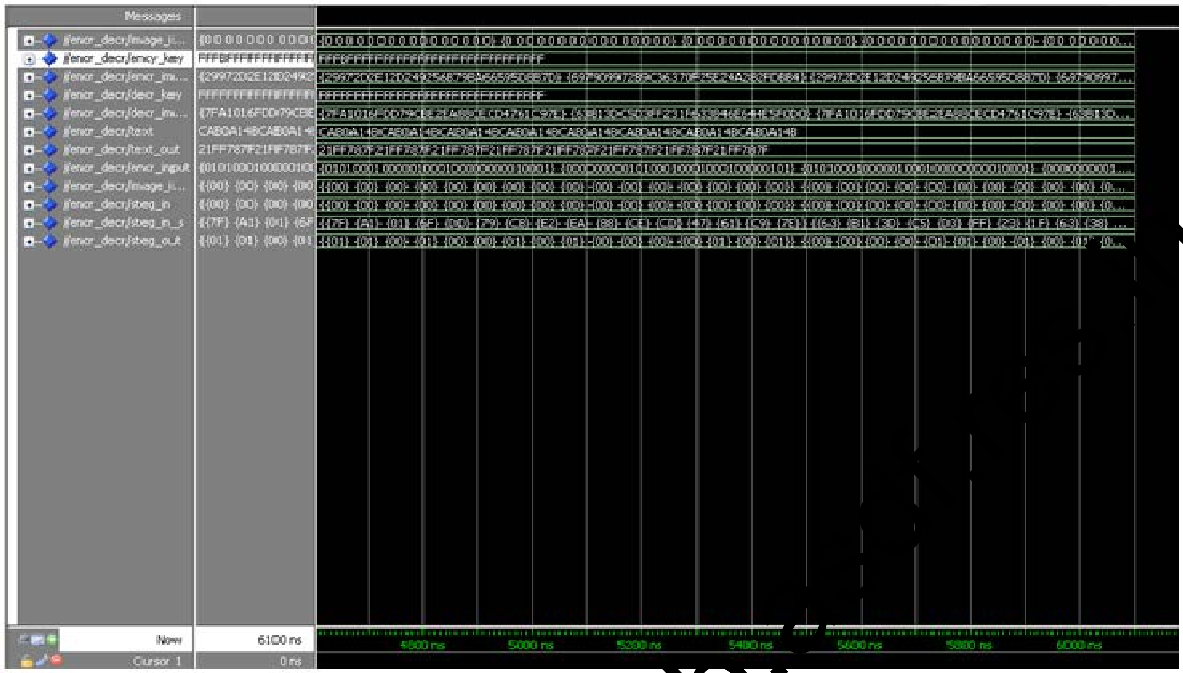
RESULT:

- If the encryption and decryption key is wrong ,then the output of hidden text will be a display unknown text.
- To view original hidden message, keys or password should be loaded correctly in both encryption and decryption side



```
Messages
/encr_decr/image_i...    {0 0 0 0 0 0 0 0 0 0 0 0 0
/encr_decr/ency_key      FFFFFFFFFFFFFFFF
/encr_decr/encr_im...    {1910707CBC4FCC7
/encr_decr/decr_key      FFFFFFFFFFFFFFFF
/encr_decr/decr_im...    {0101000100000100
/encr_decr/text          CAB0A14BCAB0A14
/encr_decr/text_out      CAB0A14BCAB0A14
/encr_decr/encr_input    {0101000100000100
/encr_decr/image_i...    {{00} {00} {00} {00
/encr_decr/steg_in       {{00} {00} {00} {00
/encr_decr/steg_in_s     {{01} {01} {00} {01
/encr_decr/steg_out      {{01} {01} {00} {01
Now      1600 ns
Cursor   0 ns
```

If the key or password loaded is wrong then the output of the decrypted emmbedded text will be wrong as follows in below simulation analysis..



```
Messages
/encr_decr/image_i...      {0 0 0 0 0 0 0 0 0 0 0 0
/encr_decr/encry_key       FFFBFFFFFFFFFFFFFF
/encr_decr/encr_im...      {29972D2E12D2492
/encr_decr/decr_key        FFFFFFFFFFFFFFFFF
/encr_decr/decr_im...      {7FA1016FDD79CBE
/encr_decr/text            CABDA148CABDA14
/encr_decr/text_out        21FF787F21FF787F
/encr_decr/encr_input      {01010001000000100
/encr_decr/image_i...      {{00} {00} {00} {00
/encr_decr/steg_in         {{00} {00} {00} {00
/encr_decr/steg_in_s       {{7F} {A1} {01} {6F
/encr_decr/steg_out        {{01} {01} {00} {01
Now                        6100 ns
Cursor 1                   0 ns
```

- ♦ In this paper i proposed a new technique of using image as a cover medium for concealing cryptographic communication.

- ♦ This is a new technique, which can be used to disguise the use of encrypted communication as well as keep the hidden information secret.

CONCLUSION

Cryptography can protect your data from thieves and impostors. You can encrypt the files on your hard disk so that even if your enemies gain physical access to your computer, they won't be able to access its data. Cryptography can make it hard to forge email and hard to read other people's messages. Steganography is a really interesting subject and outside of the main stream cryptography and system administration that most of us deal with day after day. But it is also quite real; this is not just something that's used in the lab or an arcane subject of study in academia .Although encrypted data is difficult todecipher, it is relatively easy to detect .Encryption only obscures a message's meaning, not its existence. Therefore, Steganography, a technique that hides the existence of a message ,can be used to supplement encryption. The technique can be used everywhere which requires transfer of data through network. It can be used to transfer the military messages. It can be used in banks to secure important information from intruders. It can be used by companies to secure their confidential data. It is beneficial for securely storing sensitive data, such as hiding system passwords or keys within other files.

ACKNOWLEDGMENT

I would like to thanks Mr.S.Joseph(asst.prof of cr college)

REFERENCES

[1]. Abbas Cheddad, Joan Condell, KevinCurran, Paul McKevitt (2010), Digital image steganography: Survey and analysis of currentmethods Signal Processing 90 pp727–752

[2].R.Amirtharajan,R.Akila,P.Deepikachowdavarapuohn (2010), A Comparative Analysis of
Image Steganography International Journal of Computer Applications 2(3).pp 41-47

[3]. W. Bender, D. Gruhl, N. Morimoto, A. Lu(1996), Techniques for data hiding, IBM Syst. J.
35 (3&4) pp 313–336.

[4]. BruiceSchneier, Applied CryptographyProtocols, Algorithm and Source Code in C.
Second edition. Wiley India edition 2007

[5] AES page available viahttp://www.nist.gov/CryptoToolkit.

[6] Computer Security Objects Register(CSOR): http://csrc.nist.gov/csor/.

[7] J. Daemen and V. Rijmen, AES Proposal:Rijndael, AES Algorithm Submission,September 3, 1999, available at [1].

[8] J. Daemen and V. Rijmen, The blockcipherRijndael, Smart Card research and Applications, LNCS 1820, Springer-Verlag, pp.288-296.

[9] B. Gladman‟s AES related home pagehttp://fp.gladman.plus.com/cryptography_technology/.

[10] A. Lee, NIST Special Publication 800-21,Guideline for Implementing Cryptographyin
the Federal Government, National Institute of Standards and Technology, November 1999.