International Journal of Advanced Technology and Innovative Research

ISSN 2348–2370
Vol.08,Issue.05,
May-2016,
Pages:0858-0863

www.ijatir.org

Secrete Image Transmission using AES Algorithm on Raspberry Pi

B. GURUNADH[1], T. SRAVANTHI[2], B. BRAHMAREDDY[3]

[1]PG Scholar, Dept of ECE, Vignana Bharathi Institute of Technology, Hyderabad, TS, India.
[2]Assistant Professor, Dept of ECE, Vignana Bharathi Institute of Technology, Hyderabad, TS, India.
[3]Professor & HOD, Dept of ECE, Vignana Bharathi Institute of Technology, Hyderabad, TS, India.

Abstract: Steganography means covered writing. Its goal is to hide the fact that communication is taking place. The growing possibilities of modem communications need the special means of security especially on computer network. The network security is becoming more important as the number of data being exchanged on the Internet increases. Therefore, the confidentiality and data integrity are required to protect against unauthorized access. This has resulted in an explosive growth of the field of information hiding. In addition, the rapid growth of publishing and broadcasting technology also requires an alternative solution is hiding information. The copyright of digital media such as audio, video and other media available in digital form may lead to large-scale unauthorized copying.

Keywords: Cryptography, Crypto-steganography, Steganography.

I. INTRODUCTION

With the explosive growth of internet and the fast communication techniques in recent years the security and the confidentiality of the sensitive data has become of prime and supreme importance and concern. To protect this data from unauthorized access and tampering various methods for data hiding like cryptography, hashing, authentication have been developed and are in practice today. Steganography is the method of hiding secret information like password, text, image behind another text, image and audio file. Steganography is intended to provide secrecy. Least significant bit (LSB) algorithm is used in steganography for embedding secret data into cover file. This approach is to replace the data of lower bit in a cover file by a secret data. Here We develop steganography application by using MATLAB. Select audio-video file, behind which user want to hide data. Separate audio and video from selected audio-video file using a 'Easy Audio-Video Seperator'. Save audio file as wav file. Select original video avi file. Read the file using 'Video Reader' function. Select one frame from video file here we hide the image. Behind audio we hide text file and behind video we hide image. Now combine stego audio and stego video file using 'Cute audio video merger'.

II. EXISTING METHOD

Cryptography: In existing cryptography, we can just convert plain text to chipper text with encryption. Process as shown in above diagram to do so, we need to provide one public key like any password. Then modified , that means encrypted cipher text will be forwarded to other end .At the receiver's end again we can convert this cipher text to plain text using decryption by giving public key as we have given at the transmitter end as shown in Fig.1.

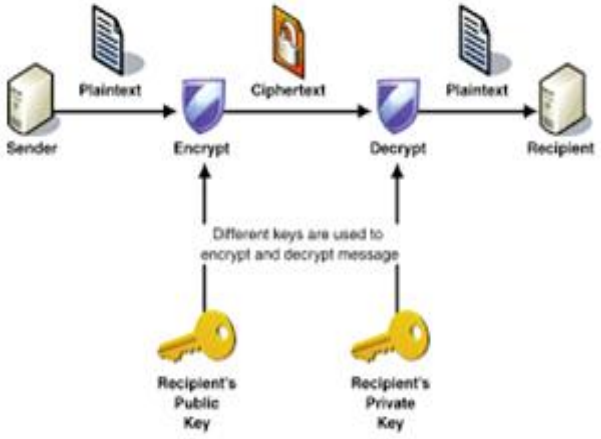

Fig.1. Block diagram of existing cryptography technique.

Steganography: Here we are using hiding technique called Steganography but the main algorithm is DES (Digital Encryption standard) as shown in Fig.2.

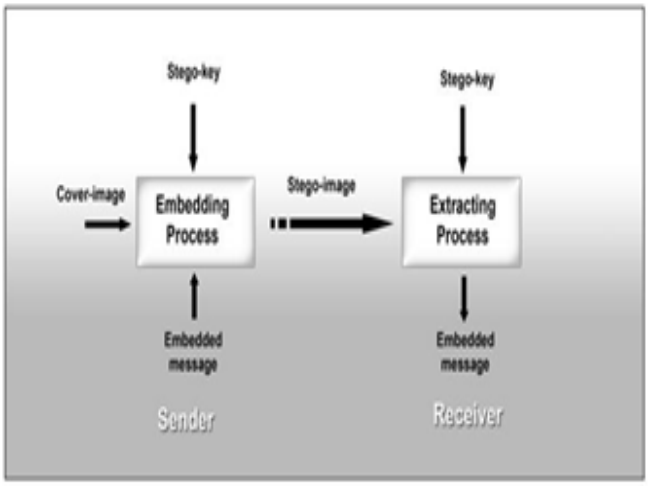

Fig.2. Block diagram of existing cryptography technique.

The actual existing Steganography looks as shown in the above diagram where we are converting embedded message to stego image. Stego-image is combination of cover-image and stego-key. The stego-key is password that we can give manually. There is no algorithm for this stego key. To form stego-image we are combining cover-image and embedded message. This will be done by algorithm called DES. In existing Steganography technique this is how we are providing security to data. But in this we cannot perform operations to audio, video, Image and text at a time. This is one drawback in this existing model.

III. PROPOSED METHOD

Here we are using Raspberry Pi board as our platform. It has an ARM-11 SOC with integrated peripherals like USB, Ethernet and serial etc. On this board we are installing Linux operating system with necessary drivers for all peripheral devices and QT software to support GUI. Here we connect mouse and keyboard to Raspberry Pi through usb host.

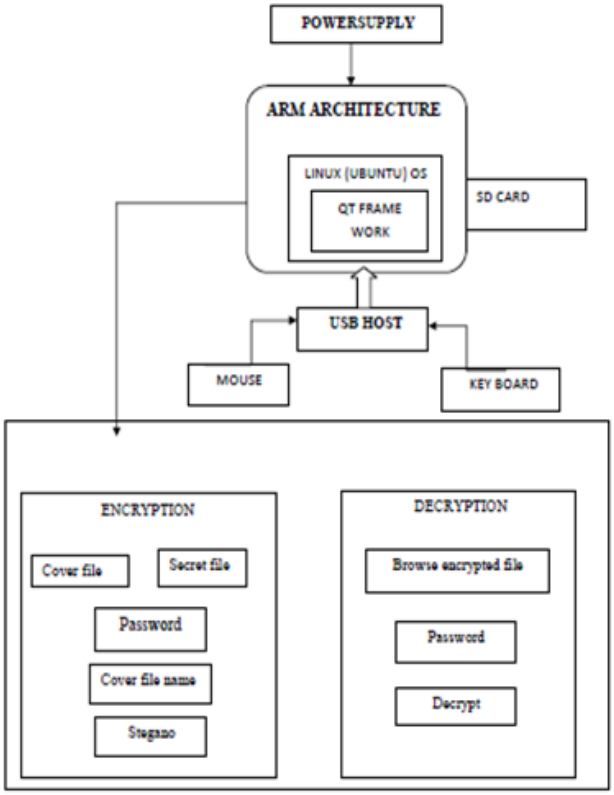

Fig.3. Block diagram of proposed system.

Here we are hiding secret information behind image or audio file. The cover file and secret file is in digital format. Here we can hide text data behind audio or image, we can hide image behind image or audio, we can hide audio behind audio file as shown in Fig.3. The size of secret data should be less than the size of cover data. Select a cover file and secret file by click on buttons displayed on LCD display. Type a password for security of the file, and name the file of the stegano file, click on stegano button. Inside code will run and hides the secret information behind cover file. Select encrypted file by cliking on the Browse encrypted file button, type the password given for encryption, and click on decrypt button it will decrypt the data. If we type a wrong password it does not decrypt the file. The system provide more security for the data. Hacker is not able to see the data behind the cover file. By using steganography there is no difference between the cover file and stego file. Our human eye is not able to percept the difference between the cover file and secret file because the lsb bit of cover file is replaced by a secret file.

A. Raspberry Pi

The Raspberry Pi is a credit-card-sized single-board computer developed in the UK by the Raspberry Pi Foundation with the intention of promoting the teaching of basic computer science in schools. The Raspberry Pi is a small, powerful and lightweight ARM based computer which can do many of the things a desktop PC can do. The powerful graphics capabilities and HDMI video output make it ideal for multimedia applications such as media centers and narrowcasting solutions as shown in Fig.4.

Fig.4. Raspberrypi board Hard ware.

The Raspberry Pi is based on a Broadcom BCM2835 chip. It does not feature a built-in hard disk or solid-state drive, instead relying on an SD card for booting and long-term storage. Raspberry Pi has a strong processing capacity because of using the ARM11 architecture and Linux-based system. In terms of control and interface, it has 8 GPIO, 1 UART, 1 I2C and 1 SPI, which are basically meet the control requirement are simple and easy-used open source peripheral driver libraries. The Raspberry Pi has a Broadcom BCM2835 system on a chip (SoC), which includes an ARM1176JZF-S 700 MHz processor, Video Core IV GPU, and was originally shipped with 256 megabytes of RAM, later upgraded to 512 MB. It does not include a built-in hard disk or solid-state drive, but uses an SD card for booting and persistent storage.

B. AES Algorithm

The Advanced Encryption Standard (AES), also referenced as Rijndael (its original name), is a specification for

the encryption of electronic data AES is based on a design principle known as a substitution-permutation network, combination of both substitution and permutation, and is fast in both software and hardware. Unlike its predecessor DES, AES does not use a Feistel network as shown in Fig.5. AES is a variant of Rijndael which has a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits. By contrast, the Rijndael specification is block and key sizes that may be any multiple of 32 bits, both with a minimum of 128 and a maximum of 256 bits.

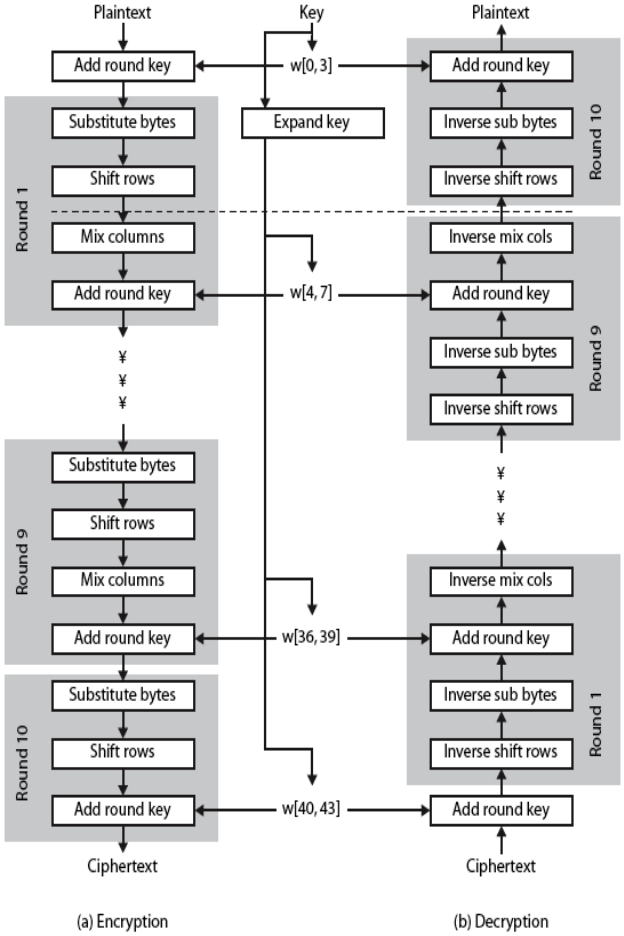

Fig.5.Steps in AES Algorithm.

AES operates on a 4×4 column-major order matrix of bytes, termed the state, although some versions of Rijndael have a larger block size and have additional columns in the state. Most AES calculations are done in a special finite field. The key size used for an AES cipher specifies the number of repetitions of transformation rounds that convert the input, called the plaintext, into the final output, called the cipher text. The numbers of cycles of repetition are as follows:

- 10 cycles of repetition for 128-bit keys.
- 12 cycles of repetition for 192-bit keys.
- 14 cycles of repetition for 256-bit keys.

Each round consists of several processing steps, each containing four similar but different stages, including one that depends on the encryption key itself. A set of reverse rounds are applied to transform ciphertext back into the original plaintext using the same encryption key.

C. The Subbytes Step

In the SubBytes step, each byte $a_{i,j}$ in the state matrix is replaced with a SubByte $S(a_{i,j})$ using an 8-bit substitution box, the Rijndael S-box. This operation provides the non-linearity in the cipher The S-box used is derived from the multiplicative invere over GF(2^8), known to have good non-linearity properties as shown in Fig.6. To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function with an invertible affine transformation. The S-box is also chosen to avoid any fixed points (and so is a derangement), i.e., $S(a_{i,j}) \neq a_{i,j}$, and also any opposite fixed points, i.e., $S(a_{i,j}) \oplus a_{i,j} \neq 0xFF$. While performing the decryption, Inverse SubBytes step is used, which requires first taking the affine transformation and then finding the multiplicative inverse (just reversing the steps).

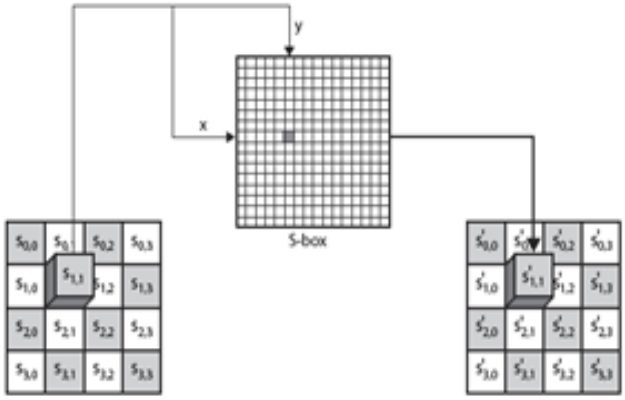

Fig.6. Sub byte Step procedure.

Sub Bytes Replacement:

TABLE I: Predefined S-Box in AES Algorithm

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

D. The ShiftRows Step

The ShiftRows step operates on the rows of the state; it cyclically shifts the bytes in each row by a certain offset. For AES, the first row is left unchanged. Each byte of the second row is shifted one to the left as shown in Fig.7. Similarly, the third and fourth rows are shifted by offsets of two and three respectively. For blocks of sizes 128 bits and 192 bits, the shifting pattern is the same. Row n is shifted left circular by n-1 bytes. In this way, each column of the output state of the ShiftRows step is composed of bytes from each column of the input state. (Rijndael variants with a larger block size have slightly different offsets). For a 256-bit block, the first row is unchanged and the shifting for the second, third and fourth row is 1 byte, 3 bytes and 4 bytes respectively—this change only applies for the Rijndael cipher when used with a 256-bit block, as AES does not use 256-bit blocks. The importance of this step is to avoid the columns being linearly independent, in which case, AES degenerates into four independent block ciphers.

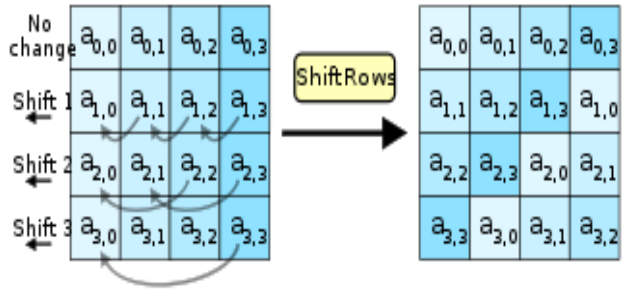

Fig.7. Shift Rows in AES.

E. The Mixcolumns Step

In the MixColumns step, the four bytes of each column of the state combined using an invertible linear transformation. The MixColumns function takes four bytes as input and outputs four bytes, where each input byte affects all four output bytes. Together with ShiftRows, MixColumns provides diffusion in the cipher. During this operation, each column is transformed using a fixed matrix (matrix multiplied by column gives new value of column in the state):

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \quad (1)$$

Matrix multiplication is composed of multiplication and addition of the entries. Entries are 8 bit bytes treated as coefficients of polynomial of order x^7. Addition is simply XOR. Multiplication is modulo irreducible polynomial $x^8+x^4+x^3+x+1$. If processed bit by bit then after shifting a conditional XOR with 0x1B should be performed if the shifted value is larger than 0xFF (overflow must be corrected by subtraction of generating polynomial). These are special cases of the usual multiplication in GF(2^8) as shown in Fig.8. In more general sense, each column is treated as a polynomial over GF(2^8) and is then multiplied modulo x^4+1 with a fixed polynomial $c(x) = 0x03 \cdot x^3 + x^2 + x + 0x02$. The coefficients are displayed in their hexadecimal equivalent of the binary representation of bit polynomials from GF(2)[x]. The MixColumns step can also be viewed as a multiplication by the shown particular MDS matrix in the finite field GF(2^8)

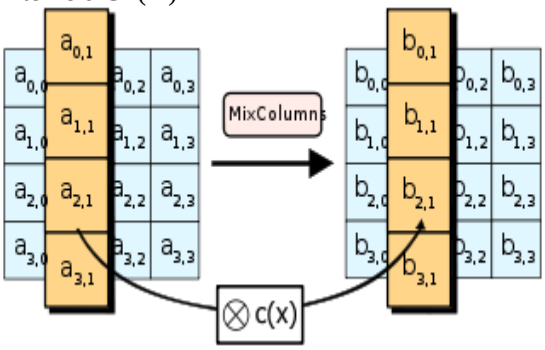

Fig.8. Mix Columns in AES.

F. The AddRoundKey Step

In the AddRoundKey step, the subkey is combined with the state. For each round, a subkey is derived from the main keyusing Rijndael's key schedule; each subkey is the same size as the state as shown in Fig.9. The subkey is added by combining each byte of the state with the corresponding byte of the subkey using bitwise XOR.

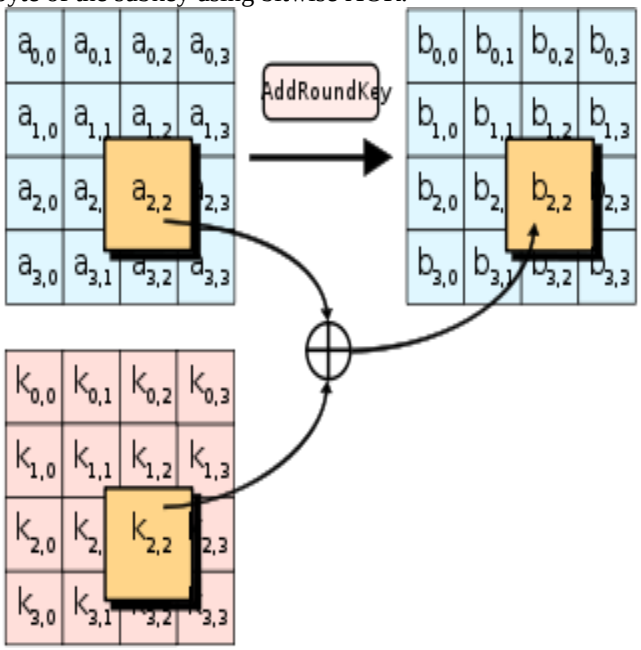

Fig.9. Add Round Key step in AES.

IV. CONCLUSION

The project "Secrete Image Transmission Using AES Algorithm on Raspberry Pi" has been successfully designed and tested. It has been developed by integrating features of all the hardware components and software used. Presence of every module has been reasoned out and placed carefully thus contributing to the best working of the unit. Secondly, using highly advanced Raspberry pi board and with the help of growing technology the project has been successfully implemented.

V. SOFTWARE IMPLEMENTATION

Software Specifications:

Operating System: Ubuntu 12.04
Language: C,C++
Platform: Qt frame work

Qt Embedded Frame Work: Qt is a cross-platform application framework that is widely used for developing application software with a graphical user interface (GUI) (in which cases Qt is classified as a widget toolkit), and also used for developing non-GUI programs such as command-line tools and consoles for servers. Qt uses standard C++ but makes extensive use of a special code generator (called the Meta Object Compiler, or moc) together with several macros to enrich the language. Qt can also be used in several other programming languages via language bindings. It runs on the major desktop platforms and some of the mobile platforms. It has extensive internationalization support. Non-GUI features Include SQL database access, XML parsing; thread management, network support, and a unified cross-platform application programming interface (API) for file handling as shown in Fig.10.

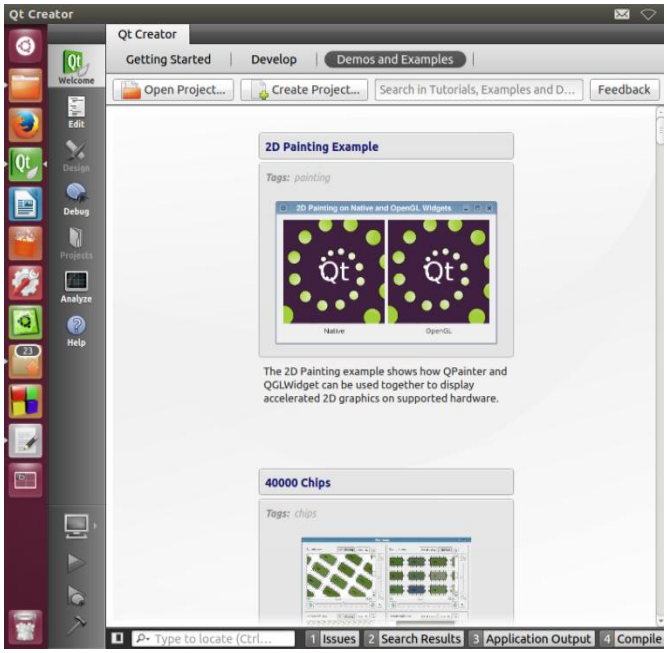

Fig.10.Qt Interactive Development Environment (IDE).

VI. HARDWARE IMPLEMENTATION

Design Aspects:

- Initially we have to enable the ARM11 board by giving the power supply.
- Connect keyboard, mouse, lcd display to ARM11.
- Open Qt software in LCD display.
- Build, compile, run the code in Qt
- Select cover file, secret file from the SDcard.
- Type the password for security.
- Click on stego button, algorithm will run and hide the secret message in cover file.
- Now file is encrypted.
- Browse the encrypted file and type the password.
- Now the file is decrypted.

Design Approach: Connections onthe Raspberry PI Board as shown in Fig.11.

Fig.11. Hardware connections with Display.

Test Plan:

- Initially power ON raspberry pi and lcd display.
- Start the os by using start_x command.
- Open terminal in raspberry pi desktop.
- Run the file in terminal of raspberry pi desktop.
- Select cover file, secret file from the SDcard.
- Type the password for security.
- Click on stego button, algorithm will run and hide the secret message in cover file.
- Now file is encrypted..
- Browse the encrypted file and type the password.
- Now the file is decrypted.

VII. REFERENCES

[1] R.J. Anderson and F. A. P. Petitcolas (2001) "On the limits of the Steganography", IEEE Journal Selected Areas in Communications, 16(4), pp. 474-481.

[2] Neil F. Johnson and SushilJajodia, "Steganalysis of Image Created Using Current Steganography Software", Workshop of Information Hiding Proceedings, Portland Oregon, USA,15-17 April,1998. Lecture Notes in Computer Science, Vol.1525, Springer-Verlag (1998).

[3] Petitcolas, F.A.P., Anderson, R. J. and Kuhn, M.G. (1999) "Information Hiding -A Survey", Proceedings of the IEEE, Special issue on Protection of Multimedia Content, vol. 87, no. 7, pp.1062-1078.

[4] J. Daemen and V. Rijmen, "AES Proposal: Rijndael. NIST AES Proposal," June 1998. Available http://csrc. nist.gov/ encryption/aes/rijndael/Rijndael.pdf.

[5] B. Pfitzmann, "Information Hiding Terminology", First International Workshop on Information Hiding,May 30 - June 1, 1996, Cambridge,UK, pp. 3 47-3 50.

[6] Shay Gueron,Intel Mobility Group Israel Development Center, Israel ,White Paper on "Advanced Encryption Standard (AES) Instructions Set".

[7]Westfield Andreas and Andreas Pfitzmann, Attacks onSteganographic Systems.Third International Workshop, IH'99 Dresden Germany, October Proceedings, Computer Science.

[8]v.sathyal,k.balasuhramaniyam2,n.murali3, .rajakumaran4, vigneswari5 "Data hiding in audio signal, video signal text and jpeg images" department of computer science and engineering, prist university, tanjore, india.

[9] Mohammad Pooyan, Ahmad Delforouzi "LSB-based Audio Steganography Method Based on Lifting Wavelet Transform"007 IEEE International Symposium on Signal Processing and Information Technology.

[10] P.P.Balguri,S.K.Jagtap, "Intelligent Processing An approach of Audio Steganography" IEEE2012 ICCICT,Oct 19-20,Mumbai,India,pp 1-6.