International Journal of Computer Science and Engineering (IJCSE)
Vol.1, Issue 1 Aug 2012 105-115
© IASET

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

A UNIFIED SCHEME OF DATA EMBEDDING THROUGH STEGANOGRAPHY AND CRYPTOGRAPHY

[1]T.SUKUMAR & [2]KR.SHANTHA

[1]AssistantProfessor, SVCE, Sriperumbudur 602 105, India

2Professor, SVCE, Sriperumbudur 602 105, India

ABSTRACT

The privacy of the data transferred between the nodes of the network is the topic which is researched by many people. However, cryptography techniques, was always been an ultimate idea to protect the data. We propose an idea using both cryptography and steganography techniques to protect the data that is to be transmitted. In existing system, permutation algorithms applied to original images encrypt and embed into image format. In proposed system, text/image format used as secret message. DES, Triple – DES and RSA permutation algorithm used for password protection to encrypt password and secret message which is converted into binary code format then it is to be embedded into carriers of media file format such as image/audio/video. Decoding is the reverse process of encoding. Keys are used to extract the secret message file from hidden file.

KEYWORDS: Data hiding, JPEG, Steganography, Cryptography.

INTRODUCTION

Steganography is an art of secure communication where the existence of the communication itself cannot be detected while steganalysis is the art of detecting the secret communication. The requirements for a “good” steganographic scheme are a high embedding capacity and secrecy. Many works has focused on how to protect private information from being attacked and/or identified.

Besides cryptography, steganography is also increasingly used for secure communication. Different from cryptography, the main goal of data hiding is to conceal the hidden data by the carrier media, so that the hidden data is transferred without drawing suspicions [1] and [2].The hiding algorithms are to maintain the natural appearance of the cover media and to keep uninvolved people from even thinking the information exists. To hide information inside an image/audio/video, there are several available domains where steganography algorithms [3] used among various types of images such that JPG, JPEG, GIF, BMP format is a commonly used standard of photographic images. In this work, a secret communication scheme is proposed, in which the data is doubly protected by both encryption stage and hiding stage. The secret message to be embedded is first processed by applying encryption techniques. Herein, the permutation algorithm that requires a pair of numbers as a key is employed to permute the original message[4]. The Encrypted data is converted into binary code format and it is embedded into the carriers of media files such as image, audio or video by managing different algorithms. The recipient performs reverse steps to extract the information: first extract the pattern of the embedded message, and then use the key which was shared previously to decipher the message. If the keys used for extraction are valid then it extracts the secret message file from hidden file [6] and [7].

Digital steganography makes use of the fact that in a number of file formats, data is reduplicated or some data is of little importance, and the hidden message does not cause noticeable changes to the file. It is used in graphics files, sound files, video, and text files, for example, image files are favored and referred to as stego-images [5]. Digital steganography is also used by people who are being censored, by governments and government agencies, by criminals, and for other reasons.

MEDIAS IN STEGANOGRAPHY A. VIDEO

In the case of transmitting large number of secret messages, steganography will not satisfy the demand. So, high embedding capacity techniques are needed. Because digital video is composed of series of frames and has greater signal space, steganography in video will get large capacity. Furthermore, with the development of multimedia and stream media on the Internet, transmitting video on the Internet will not incur suspicion. Besides, the degradation of video quality cannot be observed only by naked eyes, for it may be aroused by video compression of lower quality. These reasons make it possible for us to securely hide data in video. Steganography in video can be divided into two main classes. One is embedding data in compressed raw video, which is compressed later. The other, which is more difficult, tries to embed data directly in compressed video stream. The problem of the former is how to make the embedded message resist video compression. But because the video basically exists in the format of compression, the research of the latter is more significant. A steganography algorithm for compressed video is introduced in this paper, operating directly in compressed bit stream [9].

Audio

Audio steganography requires a text or audio secret message to be embedded within a cover audio message. Due to availability of redundancy, the cover audio message before steganography and the stego message after steganography remain the same [3].

Text

However, text steganography is considered to be the most difficult kind of steganography due to the lack of redundancy in text as compared to image or audio. However, it requires less memory and provides for simpler communication. One method that could be used for text steganography is data compression. Data compression encodes information in one representation, into another representation. The new representation of data is smaller in size. One of the possible schemes to achieve data compression is Huffman coding. Huffman coding assigns smaller length code words to more frequently occurring source symbols and longer length code words to less frequently occurring source symbols. Unicode steganography uses look alike characters of the usual ASCII set to look normal, while really carrying extra bits of information. If the text is displayed correctly, there should be no visual difference from ordinary text. Some systems, however, may display the fonts differently, and the extra information would be easily spotted [10].

EXISTING METHOD

Fig 1.1 Quality of an Image

DRAWBACKS OF AN EXISTING METHOD

This sort of system is low secured and has low embedding capacity and very low quality for extracting secret images and because of the image to image embedding format can be easily retrieved by the hackers.

PROPOSED METHOD

To overcome the above problem, we achieve an authentication through password protection mechanism which can be done through permutation algorithm such us DES, Triple – DES and RSA Further, encrypted password and secret message as converted into binary code format which can be embedded into carriers of media file format such that image or audio or video. Decoding is the reverse of encoding, which is the process of transforming information from one format into another. If the keys used for extraction are valid then it extracts the secret message file from hidden file.

SYSTEM DESIGN ENCRYPTION AND DECRYPTION PROCESS

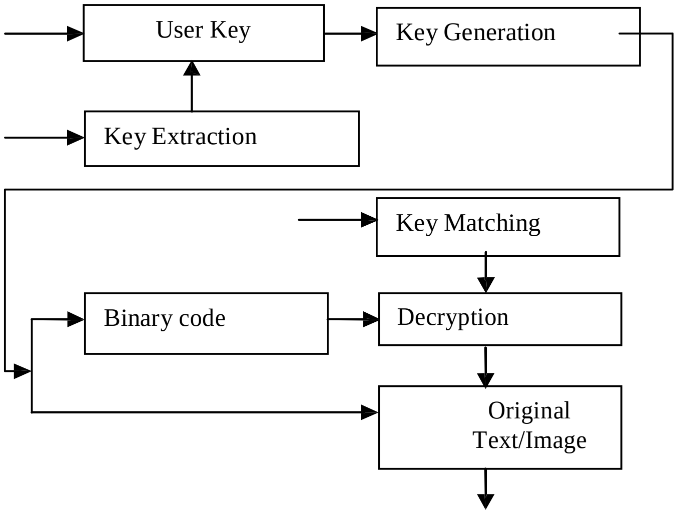
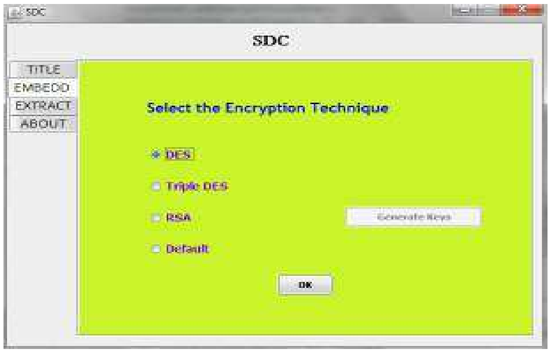


Fig 2.1 Encryption and Decryption Process

SYSTEM IMPLEMENTATION

Embedding Image on Cover Image Stage

Each pixel of the hidden image is embedded to a specific region of a JPEG image by managing the quality factors. The embedding stage is as follows. Suppose the secret message M is a binary image of size AXB, with M(A,B) = 0, corresponding to the black pixels and M(A,B) = 1, for the white pixels. The host image is of size LXW. To keep the shape of the secret image unchanged, the aspect ratio of the secret image should be identical to that of the host image (i.e., A/B = L/W). The embedding scheme includes the following two steps.



Select the Encryption Technique

● DES
○ Triple DES
○ RSA
○ Default

Fig 3.1 Selection of an Encryption Algorithm

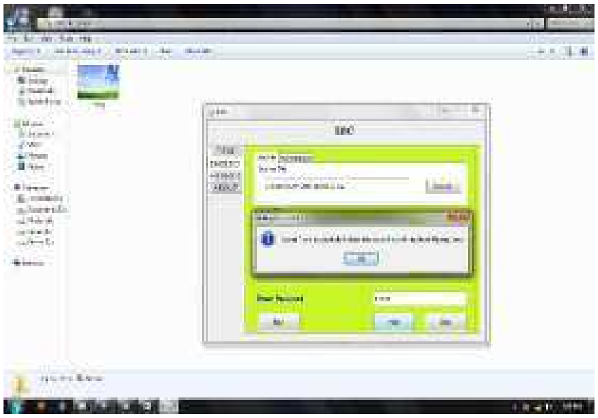


Fig 3.2 Embedding Image on Image

Extracting Image from an Image

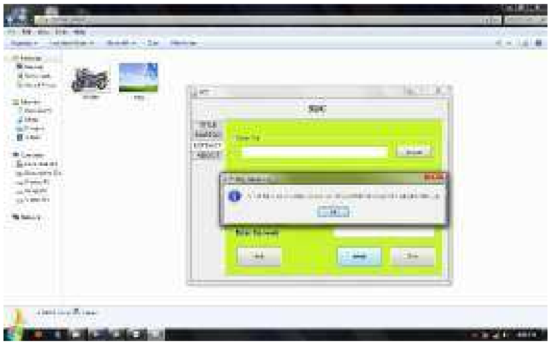


Fig 3.3 Extracting Image from an Image

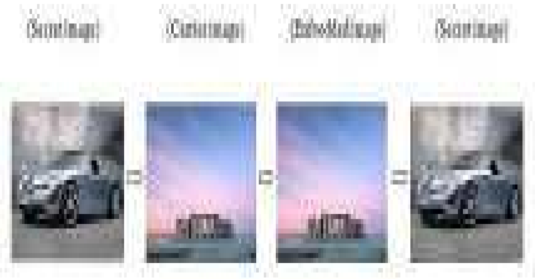


Fig 3.4 Proposed Method Image Quality

Algorithm

1. DES (Data Encryption Standard)
2. Triple – DES (Triple Data Encryption Standard)
3. RSA (Rivest – Shamir – Adleman)

DES (Data Encryption Standard)

Fundamentally DES performs only two operations on its input, bit shifting, and bit substitution. The key controls exactly how this process works. By doing these operations repeatedly and in a non-linear manner you end up with a result which cannot be used to retrieve the original without the key. Those familiar with chaos theory should see a great deal of similarity to what DES does. By applying relatively simple operations repeatedly a system can achieve a state of near total randomness. DES works on 64 bits of data at a time. Each 64 bits of data is iterated on from 1 to 16 times (16 is the DES standard). For the each iteration a 48 bit subset of the 56 bit key is fed into the encryption block represented by the dashed rectangle above. Decryption is the inverse of the encryption process. The "F" module shown in the diagram is the heart of DES. It actually consists of several different transforms and non- linear substitutions.

Triple DES (Data Encryption Standard)

Triple DES uses a "key bundle" which comprises three DES Keys, K1, K2 and K3, each of 56 bits (excluding Parity bits). The

Encryption algorithm is:

Cipher text = EK3 (DK2 (EK1 (plain text))) I.e., DES encrypts with K1, DES decrypt with K2, and then DES encrypt with K3.

Decryption is the reverse:

Plaintext = DK1 (EK2 (DK3 (cipher text))) I.e., decrypt with K3, encrypt with K2, and then decrypt with K1. Each triple encryption encrypts one block of 64 bits of data. In each case the middle operation is the reverse of the first and last. This improves the strength of the algorithm when using keying option 2, and provides backward compatibility with DES with keying option 3. Keying Options: The standards define three keying options:

Keying option 1: All three keys are independent. Keying option 2: K1 and K2 are independent, and K3 = K1.

Keying option 3: All three keys are identical, i.e. K1 = K2 = K3.

Keying option 1 is the strongest, with 3 × 56 =

168 independent key bits.

Keying option 2 provides less security, with 2 ×

56 = 112 key bits. This option is stronger than simply DES encrypting twice, e.g. with K1 and K2, because it protects against meet - in - the - middle - attacks.

RSA (Rivest-Shamir-Adleman)

The RSA algorithm involves three steps: Key generation, encryption and decryption.

Key Generation

RSA involves a public key and private key. The public key can be known to everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted using the private key. The keys for the RSA algorithm are generated the following way: Choose two distinct prime numbers p and q. Compute n = pq.

n is used as the modulus for both the public and private keys.

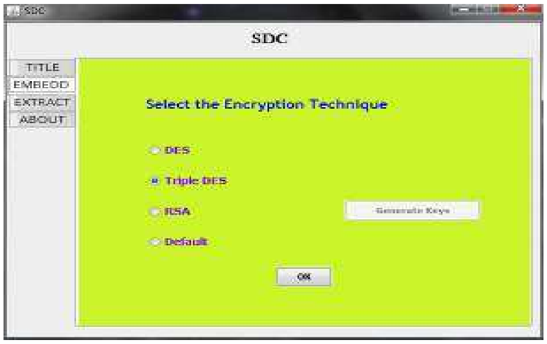
Compute $\varphi(n) = (p - 1)(q - 1)$ Choose an integer e such that $1 < e < \varphi(n)$ e = public key exponent. Small values of e have been shown to be less secure. Compute d = e–1 mod $\varphi(n)$ d is kept as the private key exponent.

Encryption: Public key (n, e) Message = m c = m e (mod n). Decryption:

d = private key exponent.

M = C D (Mod N).

Text/Image Embedding on Audio



Select the Encryption Technique

○ DES
● Triple DES
○ RSA
○ Default

Fig. 3.5 Selecting Key Encryption T-DES Algorithm

Choosing Text and Embedding on Audio

Audio is used as a carrier to embed the secret message on Text. In this process the secret message itself. Authentication can be done using password protection mechanism through T-DES algorithm. So that secret file converted into binary format and it is embedded into audio file as shown in Fig 3.6.

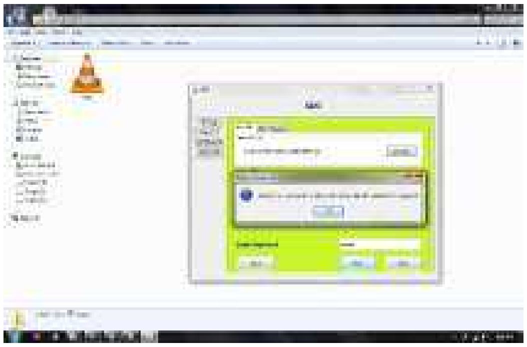


Fig.3.6 An Embedding Text on Audio

Extracting Text from Audio

User has to select the embedded file and enter the password if it is valid then extract the secret message from embedded audio file as shown in Fig 3.7. The entered password is invalid extraction process become failed. So user has been asked to enter a valid password once again to retrieve the secret message. The reverse process of embedding is to extract the binary code from embedded audio which can be converted to secret text file as shown in Fig 3.7.

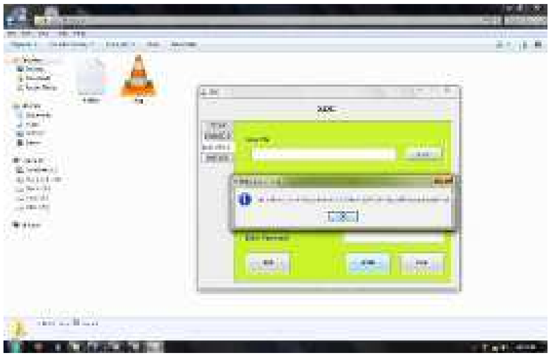


Fig.3.7 An Extracting Text from Audio

Text/Image Embedding on Video:

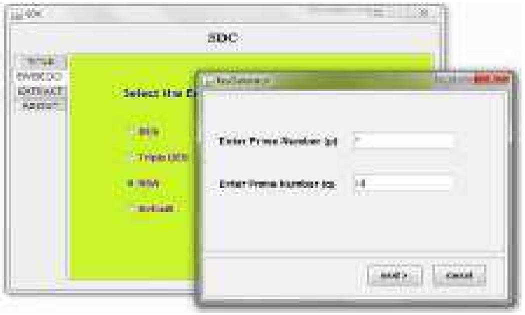


Fig.3.8 Public Key Generation

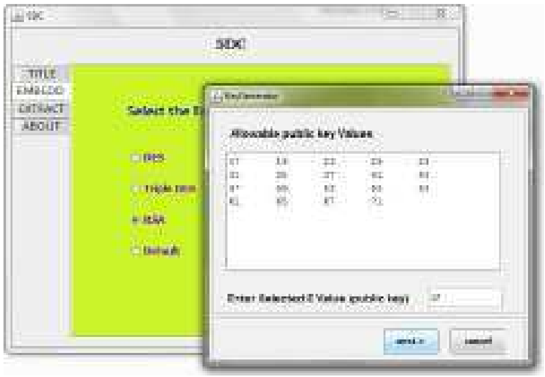
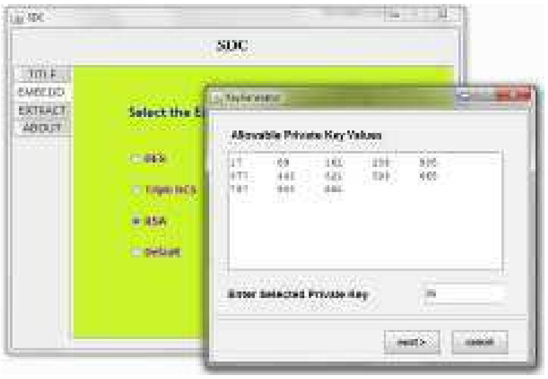


Fig.3.9 Selection of Public Key Exponent Value



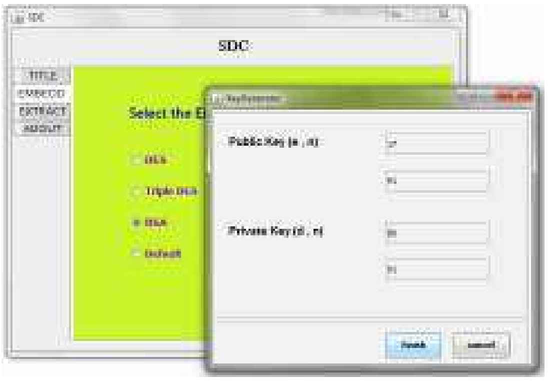
○ DES
○ Triple DES
● RSA
○ Default

Allowable Private Key Values

[illegible]

Enter Selected Private Key [illegible]

Fig.3.10 Selection of Private Key Exponent



○ DES
○ Triple DES
● RSA
○ Default

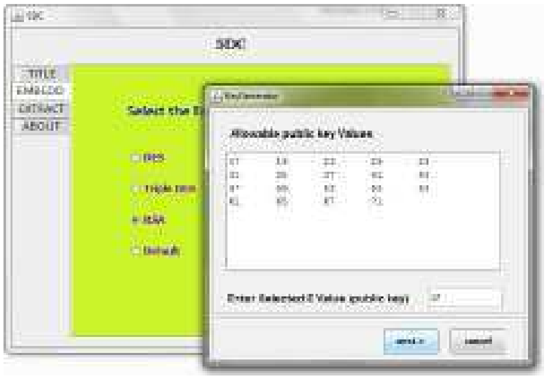
Public Key (e , n) [illegible] [illegible]

Private Key (d , n) [illegible] [illegible]

Fig 3.11 Public and Private Key Generation

Image Embed into Video

Video is used as a carrier to embed the secret message on image. In this process the secret message itself. Authentication can be done using password protection mechanism through DES algorithm. So that secret file converted into binary format and it is embed into video as Fig. 3.12.



○ DES
○ Triple DES
● RSA
○ Default

Allowable public key Values

[illegible]

Enter Selected E Value (public key) [illegible]

Fig. 3.12 Embedding Image on Video Using Public Key

Extracting Image from Video

User has to select the embedded file and enter the password if it is valid then extract the secret message from embedded video file as shown in Fig 3.13. The entered password is invalid extraction process become failed. So user has been asked to enter a valid password once again to retrieve the secret message. The reverse process of embedding is to extract the binary code from embedded image which can be converted to secret image as shown in Fig 3.13

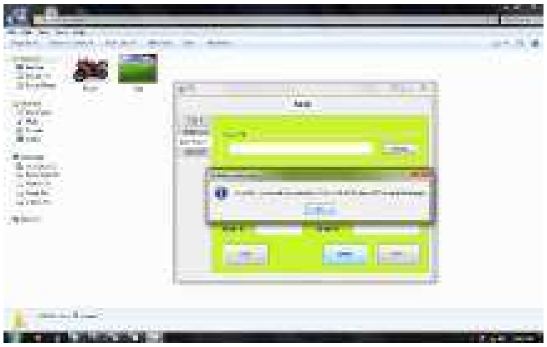


Fig.3.13 Extracting Image from Video Using Private Key

CONCLUSIONS AND FUTURE ENHANCEMENT

In this paper, we achieved secured data communication of sending an image in encrypted format using DES, Triple DES & RSA algorithms over a network and also it is embedded on various carriers such as image, audio and video file formats. In existing work, secret data communication achieved only for JPEG format and also quality is not good while retrieving a secret message. Contrasting with an existing work we were achieved a secured data Communication of an image formats such as JPEG; GIF can be embedded on above mentioned formats and also another type of carriers are supporting above mentioned process such as MP3, WAV, AVI & MPEG. Finally, we achieved good quality of an image after extraction process in a secured manner.

We focused towards to achieve security. In future work will be focused towards to achieve compression with good security and good quality of an image.

REFERENCES

1. Farid .H, “Exposing digital forgeries from JPEG ghosts,” IEEE Trans. Inform. Forensics Security, vol. 4, no. 1, pp. 154–160, Mar. 2009.

2. Gul .G and Kurugollu .F, “A novel universal steganalyser design: “LogSv”,” in IEEE Int. Conf. Image Processing (ICIP 2009), Cairo, Egypt, 2009.

3. Huang .F, Li .B, and Huang .J, “Attack LSB matching steganography by counting alteration rate of the number of neighbourhood gray levels,” in Proc. IEEE Int. Conf. Image Processing, Oct. 16–19, 2007, vol. 1, pp. 401–404.

4. Jing-Ming Guo. “Secret Communication Using JPEG Double Compression” Member, IEEE, and Thanh-Nam Le. IEEE Signal Processing Letters, Vol. 17, No. 10, October 2010

5. Ker .D, “A fusion of maximum likelihood and structural steganalysis,” in Proc. 9th Int. Workshop on Information Hiding, 2007, vol. 45, pp. 204–219.

6. Ni .Z and Shi .Y, “Robust lossless image data hiding designed for semi-fragile image authentication,” IEEE Trans. Circuits Syst. Video Technol., vol. 18, no. 4, pp. 497–509, 2008.

7. “Steganalysis using image quality metrics,” IEEE Trans. Image Process., vol. 12, no. 2, pp. 221–229, Feb. 2003.

8. D. Upham, JSteg, 1997. [Online]. Available: http://www.funet.fi/pub/crypt/steganography /jpeg-jsteg-v4.diff.gz.

9. M. Wu and B. Liu, “Data hiding in image and video: Part I—Fundamental issues and solutions,” IEEE Trans. Image Process., vol. 12, no.6, pp. 685–695, Jun. 2003.

10. T. Pevny and J. Fridrich, “Determining the stego algorithm for JPEG images,” Proc. Inst. Elect. Eng., Inf. Security, vol. 153, no. 3, pp. 77–164, 2006.