

A Secured Delivery Scheme for Images

Debnath Bhattacharyya¹, Poulami Das¹, Debashis Ganguly¹,
Swarnendu Mukherjee¹, Samir Kumar Bandyopadhyay², Tai-hoon Kim³

¹ Computer Science and Engineering Department, Heritage Institute of Technology,
Kolkata-700107, India
{debnathb,dasp88,DebashisGanguly,mukherjee.swarnendu}@gmail.com

² Department of Computer Science and Engineering, University of Calcutta,
Kolkata-700009, India
skb1@vsnl.com

³ Hannam University, Daejeon – 306791, Korea
taihoonn@empal.com

Abstract. Computer and Network security of late has become a major issue with the advent of Internet. Presently, many software tools are available to implement data security. In this paper, we have proposed a new technique to hide an image file entirely with in another image file keeping two considerations in mind which are Size and Degree of Security. At the source, the image which is to be hidden (target image) is encoded at the end of another image (cover image). Double layer security of the hidden image can be achieved (over the untrusted network) by; firstly, the starting point of encoding the image data is depended on the size of the images and it is stored within the encoded image at the end of its header information as a cipher text.; secondly, the target image is hidden behind the cover image by following our encrypted image hiding technique.

Keywords: Network, image, security, encryption, hiding.

1 Introduction

Data hiding can be defined as the process by which a message signals or image is imperceptibly embedded into a host or cover to get a composite signal. Steganography is the art and science of writing hidden messages in such a way that no one apart from the sender and intended recipient even realizes there is a hidden message. Generally, in steganography, the actual information is not maintained in its original format and thereby it is converted into an alternative equivalent multimedia file like image, video or audio which in turn is being hidden within another object. This apparent message (known as stego-object in usual terms) is sent through the network to the recipient, where the actual message is seperated from it.

In this paper, we have considered some important features of data hiding. Our first consideration is that of embedding information into image, which could survive attacks on the network. Next, a hybrid digital embedding technique is proposed for

hiding an image into another image in such a way that the quality of the recovered image improves significantly. Also to make the proposed scheme to run free of size constraints, we have introduced the concept of Padding before doing the actual hiding.

Another important feature, conceived in the encryption process of the images is Multilayered Security. Here, firstly the size difference of the two considered images (target and cover image) is hidden in an encrypted format with in the cover image. After that the entire target image is hidden at the end of the cover image by following our encryption technique which is described in the section-III.C. Thus if the ultimate object to be transferred gets snooped from the information channel, then also from it no way the information can be retrieved.

Lastly, in most of the algorithms designed based on the principle of Steganography, requires the sending original cover image along with the encoded cover image to the receiver. This approach makes the designed algorithm weaker as it conveys some idea of data hiding to the eavesdropper. But our method covers this lack as here only the encoded image will be sent to the receiver.

To the best of our knowledge, this work is specifically focused on protection of any information which is in the form of image. The design of this technique is based on extensive analytical as well as experimental modeling of the data-hiding process.

2 Related Works

The majority of today's steganographic systems uses images as cover media because people often transmit digital pictures over email and other Internet communication. Several methods exist to employ the concept of Steganography as well as plenty algorithms have been proposed in this regard. To gather knowledge regarding our approach, we have concentrated on some techniques and methods which are described below.

The next interesting application of steganography is developed by Miroslav Dobsicek [1], where the content is encrypted with one key and can be decrypted with several other keys, the relative entropy between encrypt and one specific decrypt key corresponds to the amount of information.

Min Wu and Bede Liu, June, 2003, proposed [2] a new method to embed data in binary images, including scanned text, figures, and signatures. The method manipulates "flippable" pixels to enforce specific block based relationship in order to embed a significant amount of data without causing noticeable artifacts. They have applied Shuffling before embedding to equalize the uneven embedding capacity from region to region. The hidden data can then be extracted without using the original image, and can also be accurately extracted after high quality printing and scanning with the help of a few registration marks.

F. Bartolini, A. Tefas, M. Barni and I. Pitas discussed the problem of authenticating video surveillance image. After an introduction used to stimulate the need for a watermark-based authentication of VS (Video Surveillance) sequences, a brief survey of the main watermark-based authentication techniques has been

presented and the requirements that an authentication algorithm should satisfy for VS applications are discussed. A novel heuristic approach which is suitable for VS visual data authentication has been proposed [3].

Mark A. Masry, 2005, proposed a novel blind watermarking algorithm designed for map and chart images. The algorithm segments the image into homogeneous regions and adds multiple watermark signals to the locations of the pixels on the boundary of several regions. The presence of these signals in the watermarked image is determined using a correlation based detector. The watermarks can be detected in the presence of synchronization errors such as those incurred by cropping the image, or shifting by several columns or rows, and in the presence of noise. The algorithm is designed to efficiently process typical map images [4].

Rehab H. Alwan, Fadhil J. Kadhim, and Ahmad T. Al-Taani, 2005, have explained a method with three main steps [5]. First, the edge of the image is detected using Sobel mask filters. Second, the least significant bit LSB of each pixel is used. Finally, a gray level connectivity is applied using a fuzzy approach and the ASCII code is used for information hiding. The prior bit of the LSB represents the edged image after gray level connectivity, and the remaining six bits represent the original image with very little difference in contrast. The given method embeds three images in one image and includes, as a special case of data embedding, information hiding, identifying and authenticating text embedded within the digital images.

Yusuk Lim, Changsheng Xu and David Dagan Feng, 2001, described the web-based authentication system consists of two parts: one is a watermark embedding system and the other is authentication system. In case of watermark embedding system, it is installed in the server as application software that any authorized user, who has access to server, can generate watermarked image. The distribution can use any kind of network transmission such as FTP, email etc. Once image is distributed to externally, client can access to authentication web page to get verification of image [6].

In 2007, Nameer N. EL-Emam proposed an algorithmic approach to obtain data security using LSB insertion steganographic method. In this approach, high security layers have been proposed through three layers to make it difficult to break through the encryption of the input data and confuse steganalysis too [7].

Unfortunately, modifying the cover image changes its statistical properties, so eavesdroppers can detect the distortions in the resulting stego-image's statistical properties. In fact, the embedding of high-entropy data (often due to encryption) changes the histogram of colour frequencies in a predictable way. So, in order to obtain more security in our prescribed method, we have embedded an entire image behind another image by modifying discrete zone of pixels. By selecting discrete zone, we have tried to avoid any remarkable change in the cover image.

3 Our Work

Before going into the details of the algorithm proposed here for invisible watermark of the information behind the cover object, it is better to mention about the selection of the images and information which are to be steganographed. Here in this paper, the

3.2 PAD (PICTURE, SIZE)

This function is used in the algorithm to pad an image to obtain an image of the desired size from the input image.

Arguments: This function will take the image, which has to be padded along with the desired image size which is to be obtained after padding.

1. Obtain the width, length of the pixel matrix of the PICTURE (say R_p and C_p).
2. Fill pixels with white color until $R_p * C_p * \text{DPI} \geq \text{SIZE}$.
Here DPI stands for Depth per Index.
3. Return the PICTURE.

3.3 SDSI_ENC (PICTURE_1, PICTURE_2, SIZE)

This function is used in the algorithm to encrypt an image with the help of either same image (self encryption) or another image to obtain an encrypted image of the desired size.

Arguments: This function will take the cover image (PICTURE_1) in which another image will be hidden; the target image (PICTURE_2), which will be hidden, and finally it will output the stego-image as final image.

1. Read the bytes from starting of the PICTURE_1.
2. Repeat step 1 if numbers of bytes read is not equal to 54 (header size for a BMP image).
3. Obtain the octal format of SIZE and count the number of digits present in that format and store it in a variable say key.
4. Read the next byte of PICTURE_1 and replace it with key.
5. Next read key number of bytes of PICTURE_1 and replace each of them with the digits present in the octal format of SIZE starting from Left.
6. Read the first byte of the PICTURE_2.
7. If the value of the read byte is 0 then replace the current byte of the PICTURE_1 with 255 and read the next byte of PICTURE_1 and PICTURE_2. Otherwise go to the next step.
8. If the value of the read byte is 255 then replace the current byte of the PICTURE_1 with 0 and read the next byte of PICTURE_1 and PICTURE_2. Otherwise go to the next step.
9. If the value of the read pixel is greater than 0 and less than 255 then do the following operations. $\text{byte}'' = \text{byte}' + \text{SPN}$ where byte' is the corresponding byte of the PICTURE_1 and SPN is the sum of prime numbers starting from 0 to SIZE.
10. Now if the calculated byte'' is greater than 255 then calculate: $\text{byte}''' = \text{byte}'' - 255$ and replace the current byte of PICTURE_1 with byte''' . Read the next byte of PICTURE_1.
11. Read the next byte of PICTURE_2 and go to step 4.
12. Now if the condition of the step 10 is false then replace the current byte of PICTURE_1 with byte'' and read the next byte of PICTURE_1 and go to step 11.

4.1 Complexity analysis of the stated algorithm

For Padding (Section III.B): To pad an image row wise, then we $O(n^2)$ is incurred. Again to pad an image column wise, complexity of $O(n^2)$ has incurred. So, overall time complexity becomes $O(2*n^2)$.

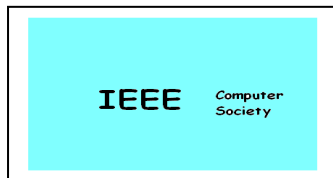
For Encryption and Decryption (Section III.C and Section III.D): For each row wise and column wise scan is being done. So, each one requires time of $O(\text{rowsize}*\text{columnsize}) \equiv O(n^2)$.

In this algorithm, operation is being done byte wise. So, no need of remembering whole pixel matrix is required. Thus amount of space required to run this algorithm comes under $O(\log n)$ and thereby it becomes an inplace algorithm.

4.2 Test Results

To test the algorithm, we have chosen one cover image having size 350000 Bytes and one target image having size 97300 Bytes and they are shown in Figure 1. Now clearly size of the cover image is greater than the size of the target image. So, calling of the PAD module of our algorithm is not required. Next, we have calculated the size parameter of our algorithm and here it is 253700 Bytes. So, according to the algorithm the value of key is 6 and the octal representation of the parameter size is 757404. That means 55th no. byte will be modified with key and stating from 56th no. byte to (key+55) no. byte will be modified by the digits present in the octal format of size starting from left.

In next, we have to move to the byte position size of the cover image and then we have to modify the successive bytes depending on the target image.



Cover Image [351000 Bytes]



Target Image [97300 Bytes]

Fig. 1. Cover Image and Target Image

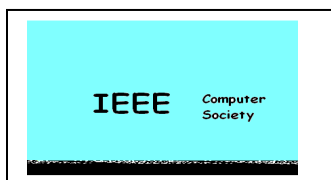


Fig. 2. Final Image



Fig. 3. Retrieved Target Image [97300 Bytes]

