Volume 3, Issue 4, April 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

# A New Approach to Hide Text in Images Using Steganography

**Vipul Sharma**
*School of Computer Science & Engineering*
*Bahra University, Shimla Hills, India*

**Sunny Kumar**
*School of Computer Science & Engineering*
*Bahra University, Shimla Hills, India*

***Abstract – In this paper, we have proposed a new steganographic algorithm that is used to hide text file inside an image. In order to increase/ maximize the storage capacity we have used a compression algorithm that compresses the data to be embedded. The compression algorithm we have used works in a range of 1bit to 8 bits per pixel ratio. By applying this algorithm we have developed an application that would help users to efficiently hide the data.***

***Keywords – Steganography, cryptography, secret key, LSB, embedding, extraction.***

## I. Introduction

Steganography is defined as the art and science of writing hidden messages in such a way that no one else, apart from the intended recepient knows the existence of the message. The word "steganography" is basically of Greek origin which means "hidden writing". The word is classified into two parts: steganos which means "secret" and "graphic" which means "writing". However, in hiding information, the meaning of steganography is hiding text or secret messages into another media file such as image, text, sound or video.**[3][4][8]** The word "steganography" is often considered similar to "cryptography" and "watermarking". Whilst watermarking ensures message integrity and cryptography scrambles the message, steganography hides it.

History of steganography dates back to 440 B.C. this technique was initiated by ancient Greeks, they shave the heads of their slaves and write the messages on their heads, after the hair had grown back, the slaves were sent to their allies without the enemies knowledge **[8].** Steganography was also used by Germans during the World War I and II. Also during the American Revolution, invisible ink was used by the revolutionaries for communication purposes. The motto behind developing steganographic methods is its application in secret communication between the members of an organization involved in mission critical situations like wars; also it can be used for communication between intelligence agencies etc.

The primary objective of steganography is to avoid drawing attention to the transmission of hidden information. If suspicion is raised, then this objective that has been planned to achieve the security of the secret message because if the hackers noted any change in the sent message then this observer will try to know the hidden information inside the message. **[1][2].**

The basic terminologies used in the steganography systems are: the cover message, secret message, the secret key and embedding algorithm **[5].** The cover message is the carrier of the message such as image, video, audio, text or some other digital media. The secret message is the information which is needed to be hidden in the suitable digital media. The secret key is usually used to embed the message depending on the hiding algorithms. The embedding algorithm is the way or the idea that usually used to embed the secret information in the cover message **[8][9].**

In steganography, before the hiding process, the sender must select an appropriate message carrier, an effective message to be hidden as well as a secret key used as a password. A robust steganographic algorithm must be selected that should be able to encrypt the message more effectively. The sender then may send the hidden message to the receiver by using any of the modern communication techniques. The receiver after receiving the message decrypts the hidden message using the extraction algorithm and a secret key **[8][9].** This paper proposes a new algorithm to hide data inside an image using steganographic technique. The algorithm that we have proposed is an enhanced version of LSB technique, that is not very much robust. Also we have implemented a compression technique to increase the hiding capacity. This all is demonstrated using an application we have build in java.

The rest of the paper is organized as follows: Section 2 would be presenting the proposed algorithm. The implementation of the system is discussed in section 3. Discussion of various results obtained from testing the system based on the proposed algorithm, with various sizes of data is explained in section 4 and finally we would be concluding the paper along with future scope.

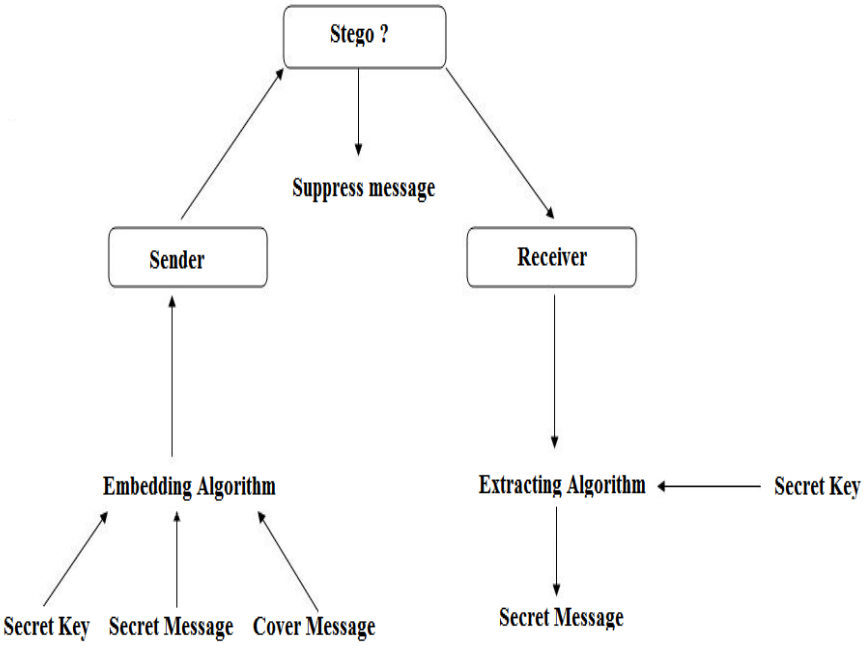



**Figure 1: General Steganographic Approach.**

## II. Proposed Algorithm

The algorithm that we have proposed in this system is basically an extension of the original LSB which is quite vulnerable. Instead of hiding the data in least significant bits of the RGB components of a pixel, we in this algorithm, would be hiding data as shown below: -

Let the data to be hidden is word "ABC"

ASCII code of A= 65 and corresponding binary is 01000001.

ASCII code of B= 66 and corresponding binary is 01000010.

ASCII code of C= 67 and corresponding binary is 01000011.

Let the first pixel's RGB component be: -

**Original red component**

| Red | | | | | | | | Green | | | | | | | | Blue | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |

Red component is replaced with binary of 65 i.e. A.

**Replaced red component**

| Red | | | | | | | | Green | | | | | | | | Blue | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |

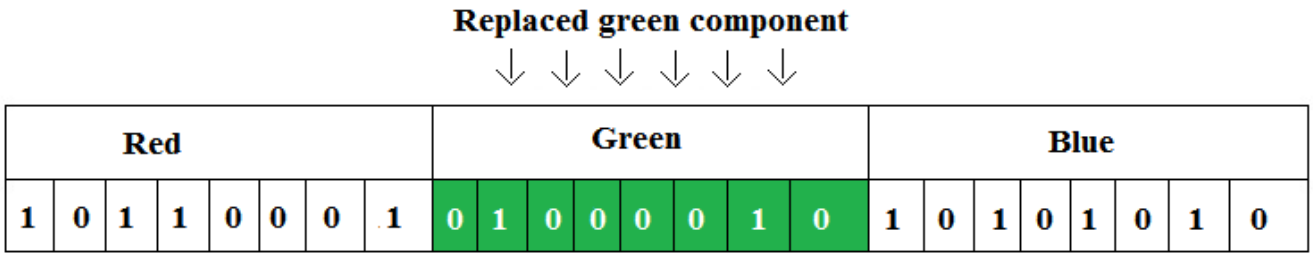
Let the second pixel's RGB component be: -

**Original green component**

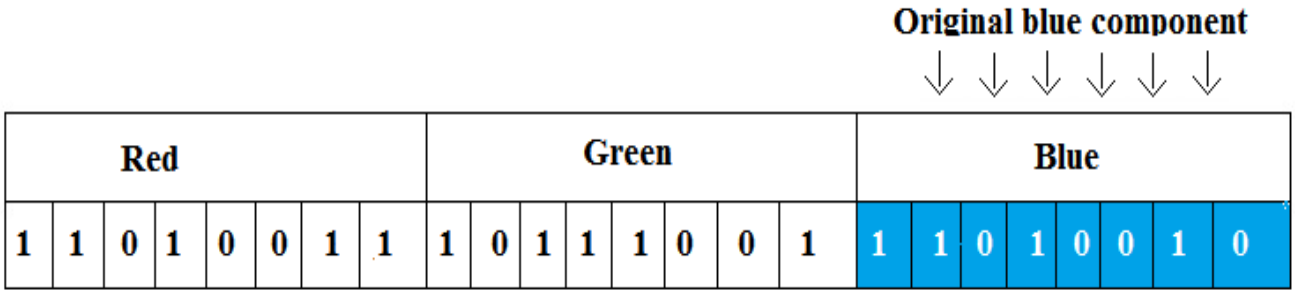
| Red | | | | | | | | Green | | | | | | | | Blue | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

Green component of second pixel is replaced with binary of 66 i.e. B.

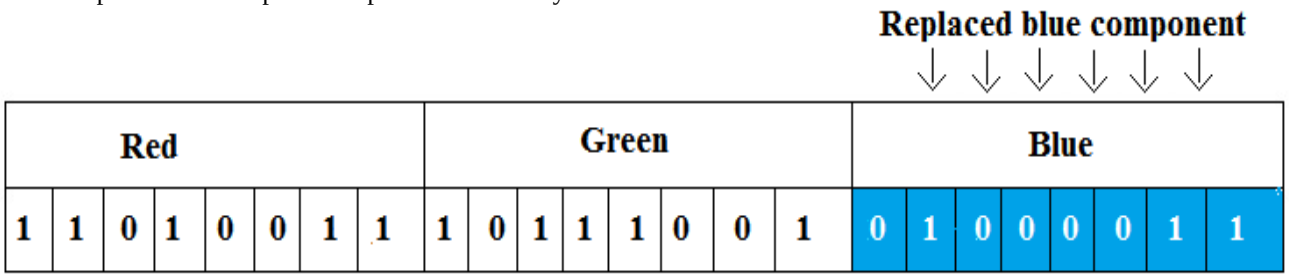
 




Let the third pixel's RGB component be: -




Blue component of third pixel is replaced with binary of 67 i.e. C.




And the process continues.

The resulting stego image that we are obtaining after the algorithm completes its execution, is distorted and is easy to detect, that some kind of alteration has been done to the image. So, to enhance the security of the secret message we would be covering the resulting stego image with a new cover image, this is the first level of security. By just looking at the resulting image no one would be able to predict that something is hidden inside it. The new cover image can be the same or different than the original.

In order to increase the storage capacity of the image, a compression algorithm has been used, we know that each component of an RGB pixel is represented with 8 bits. So, the maximum compression would be 8 bits per pixel and minimum would be 1 bit per pixel.

The proposed steganographic algorithm comprises of two embedding techniques they are data hiding technique and data retrieving technique. Data hiding technique as the name suggests is used to hide secret message and key in the cover image, while data retrieving technique is used to retrieve the key and the hidden secret message from the stego image. Therefore data is protected in image without revealing to unauthorized party.

***A. Proposed embedding technique.***

Inputs: - Text file, cover image 1, cover image 2 and secret key.

Output: - Stego image.

**Begin**

1. Select a text file, convert it into binary form and calculate the number of bits in it.
2. Select a carrier image (cover image 1) for hiding purpose, find the number of pixels, convert it into RGB image and calls the compression function.
3. If bits calculated are compatible with the image resolution, then

**Start sub iteration 1**

Replace red component of the first pixel with first character.

Replace green component of the second pixel with second character.

Replace blue component of the third pixel with third character.

And repeat iterations until pixels get exhaust.

**Stop sub iteration 1**

Else

**Repeat sub iteration 1**

Finds necessary compression ratio and perform sub iteration 2.

**Sub iteration 2**

Replace necessary bits as defined by the compression ratio in immediate component of each pixel.

Store the information about bits embedded in a binary address file.

**Stop sub iteration2**

4. Provide a security key as encryption completes.
5. Select 2nd cover image to hide the distorted stego image.

**End**

***B. Proposed extraction technique.***

Input**:** - Stego image and secret key.
Output: - Secret text file.

**Begin**

1. Browse the stego image.
2. Choose the folder in which you want to extract the hidden text file.
3. Provide necessary security key.
4. Convert the binary file into human readable form.

**End**

The main focuses of this proposed steganographic technique is to hide text files in images, compresses the text files so as to increase the overall storage capacity, applying a secret key on the resulting stego image and transferring the secret message without any vulnerability and threat.

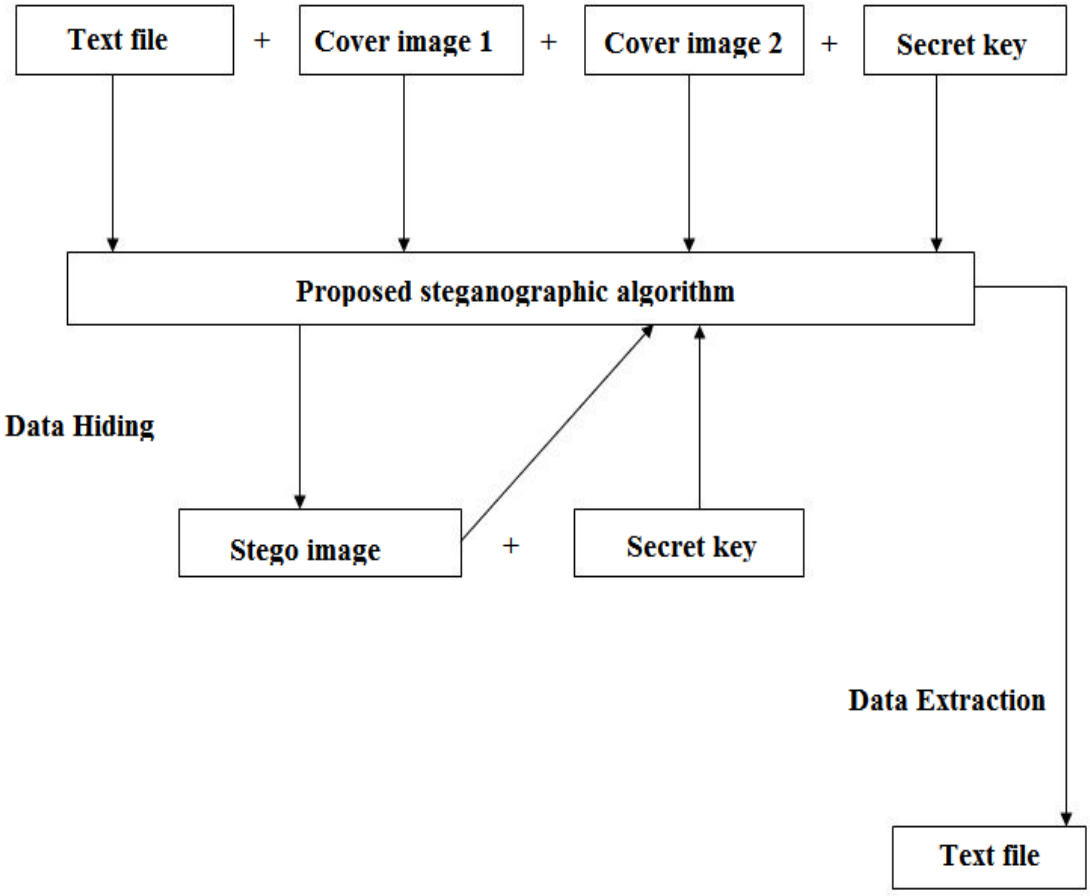



**Figure 2: General Layout of Proposed System.**

The system that we have developed in java is able to maintain the accuracy, confidentiality of the data. The system is able to hide the text files in images using a secret key and also is able to retrieve the data back from the stego image.

## III. Software Implementation

Based on the proposed algorithm, we have developed a system in Java that implements the algorithm.
Reasons for choosing java as a programming language are as: -

1. Simple to use.
2. Trustworthy in nature.
3. Purely object oriented.
4. Lots of information available online.

System we have developed basically comprises of two main interfaces, one for embedding purpose and other for the extraction process.
Overview of the system

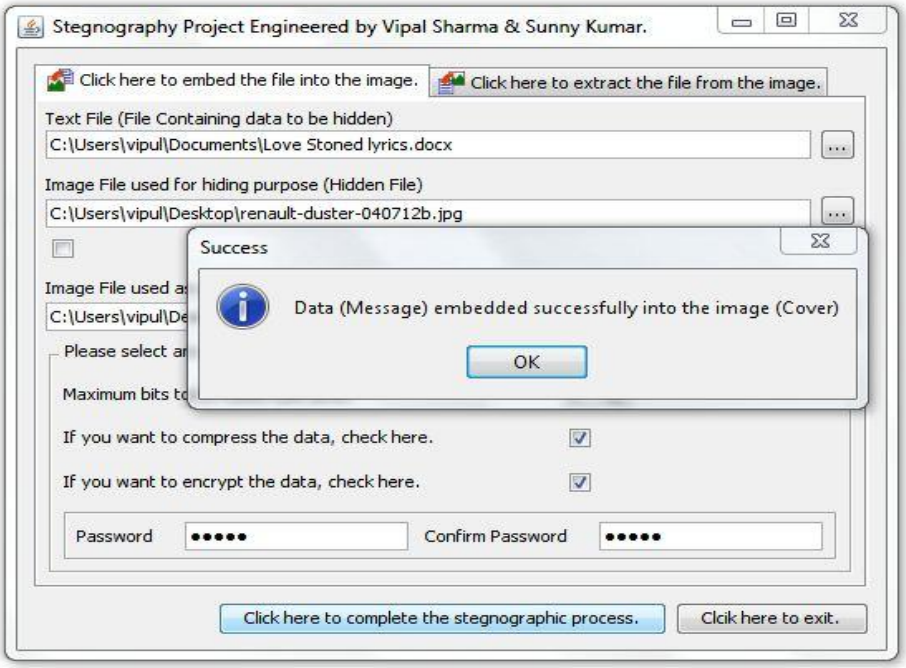



**Figure 3: Embedding Form of Our Application.**

The embedding form shown above comprises of three main browsing fields. One for the text file to be embedded, second for the image in which to embed the file and third for the cover image to hide the underlying distortion. One important point to note here is that the cover file can or cannot be same as the one used for the hiding process. After filling these necessary fields, the next step is to check the encryption checkbox. User need not to worry about the underlying compression procedure, which in turn is automatically performed by the system itself. User then need to provide the secret key twice for the verification procedure, various validations are applied here. The secret key along with the text file is embedded inside the image. Once the data has been key in and the secret key has been entered, the new stego image can be saved to a different image location. The new stego image can then be used by the user to send it via internet or email to other parties without revealing the secret data inside the image.If the other parties want to extract the hidden data from the stego image, they need to upload the new stego image using the system itself to retrieve the text file hidden inside the image by providing the secret key.
The extraction process is shown in the figure below

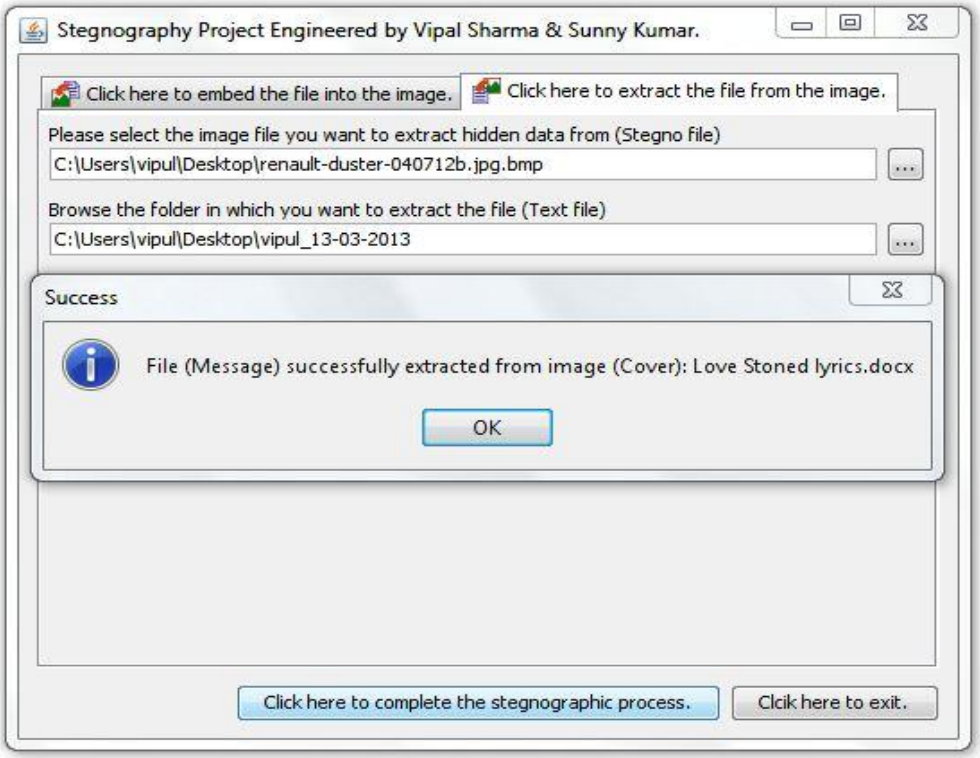



**Figure 4: Extraction Form of Our Application.**

## IV. Results And Discussion

The system is tested using the images shown in figures 5 - 7

Example 1.


**Figure 5a) Original image (.jpg)**


**Figure 5b) Cover image (.jpg)**


**Figure 5c) Stego image (.jpg.bmp)**

Fig. 5a) shows the original image before the message is stored in it. Fig. 5b) shows the cover image. Here it should be noted that the original image and the cover image are exactly same having same extensions of .jpg. The resulting stego image has a double extension of .jpg.bmp. We found that the stego image Fig. 5c) does not have any noticeable changes in it as seen from naked eyes. Also we found that the size of the stego image is higher than the original image.

Example 2.

**Figure 6a) Original image (.jpg)**

**Figure 6b) Cover image (.jpg)**

**Figure 6c) Stego image (.jpg.bmp)**

In this example hidden image and cover image Fig. 6a) and 7a) respectively are exactly the same having same extensions of .bmp. The resulting stego image Fig. 6c) obtained does not have any noticeable changes and it is found that it is having an extension of .bmp only and is larger in size than the original one.

Example 3.

**Figure 7a) Original image (.jpg)**

**Figure 7b) Cover image (.jpg)**

**Figure 7c) Stego image (.jpg.bmp)**

In this example Fig. 7a) shows the original image before the message is stored in it. Fig. 7b) shows the cover image. Here it should be noted that the original image and the cover image are different yet they have the same extension of .jpg. The resulting stego image Fig. 7c) is similar to the cover image but it's extension is.jpg.bmp and is larger in size than the original image.

Actually what is happening here is that, the data is embedded inside the original image using the algorithm we have proposed but the images we are obtaining after the embedding process, are distorted so, in order to overcome this limitation we are covering the distorted image using a cover image.

Using the proposed algorithm we have tested several sizes of .bmp and .jpg images to see various sizes of data being stored in the image.

***A. Table 1: Comparison of different file sizes in bit map format.***

| S.no. | Original image size | Cover image size | Text file size | Stego image size | Embedding | Extraction |
|---|---|---|---|---|---|---|
| 1. | 7.91 MB | 7.91 MB | 142 KB | 7.91 MB | Done | Done |
| 2. | 791 KB | 791 KB | 250 KB | 791 KB | Done | Done |
| 3. | 2.14 MB | 2.14 MB | 292 KB | 2.14 MB | Done | Done |
| 4. | 4.11 MB | 4.11 MB | 149 KB | 4.11 MB | Done | Done |
| 5. | 1.73 MB | 1.73 MB | 11.4 KB | 1.73 MB | Done | Done |

Here cover image and original image are exactly of same size and having same format of .bmp. Text files used are all word documents having extension .doc.

***B. Table 2: - Comparison of different file sizes in Jpeg format.***

| S.no. | Original image size | Cover image size | Text file size | Stego image size | Embedding | Extraction |
|---|---|---|---|---|---|---|
| 1. | 699 Kb | 1.02 MB | 409 KB | 7.91MB | Done | Done |
| 2. | 387 KB | 521 KB | 4.57 MB | 7.91 MB | Done | Done |
| 3. | 662 KB | 633 KB | 2.20 MB | 7.91 MB | Done | Done |
| 4. | 244 KB | 647 KB | 5.39 MB | 4.11 MB | Done | Done |
| 5. | 929 KB | 685 KB | 803 KB | 7.91 MB | Done | Done |

Here cover image and original images are different and are of .jpg format. Here first three text files are word documents having extension .doc. second last file is a powerpoint document and the last one is a pdf file.

From the above two tabular results we found that the case where we have used bmp images, the size of the resulting stego images is exactly the same as that of the original image. Whereas in the case where jpg format is used size differences are very large.

***C. Table 3: - Text file formats supported by our system.***

| Text file formats | Embedding | Extraction |
|---|---|---|
| .txt | Done | Done |
| .docx | Done | Done |
| .pdf | Done | Done |
| .ppt | Done | Done |
| .cpp | Done | Done |

## V. Conclusion

This paper proposed a new steganographic algorithm for hiding text files in images. Here we have also used an underlying compression algorithm with maximum compression ratio of 8 bits/ pixel. We have developed a system in java based on the proposed algorithm. Here we have tested few images with different sizes of text files to be hidden and concluded that the resulting stego images do not have any noticeable changes. Also we found that for .bmp images this algorithm works very efficiently. Hence this new steganographic approach is robust and very efficient for hiding text files in images.

## VI. Future Scope

Steganography will continue to increase in popularity over cryptography. As it gets more and more advanced as will the steganalysis tools for detecting it. At the time though most of the tools can detect the files hidden in any image. It is well accepted though, small sentences and one-word answers example a 'yes' are virtually impossible to find. This could be an area for further advances as possible compression sizes decreases further. There also seems very little in terms of tools for hiding data in videos. There are some for audio, but this is still an area, which lags behind image steganography. The future may see audio files and video streams that could possibly be decoded on the fly to form their correct messages.

**References.**

[1] H. Wu, H. Wang, C. Tsai and C. Wang, Reversible image steganographic scheme via predictive coding. 1 (2010), ISSN: 01419382, 35-43.

[2] J, Corporation, Steganography. http://www.webopedia.com/ TERM/S/steganography.html. 2005.

[3] B. Dunbar. A detailed look at Steganographic Techniques and their use in an Open-Systems Environment, Sans Institute, 1(2002).

[4] C. Christian. An Information-Theoretic Model for Steganography, Proceedings of 2nd Workshop on Information Hiding, MIT Laboratory for Computer Science. 1998.

[5] N Ghoshal, J K Mandal .A steganographic scheme for colour image authentication (SSCIA), Recent Trends in Information Technology ICRTIT 2011 International Conference on (2011), 826 831.

[6] M. D. Swanson, B. Zhu and A. H. Tewfik, Robust Data Hiding for Images, IEEE Digital Signal Processing Workshop, University of Minnesota, September 1996 ( 37-40).

[7] N. Johnson, Digital Watermarking and Steganography: Fundamentals and Techniques , The Computer Journal. (2009)

[8] N. Johnson, Survey of Steganography Software, Technical Report, January 2002.

[9] W, Peter. Disappearing Cryptography: Information Hiding: Steganography & Watermarking (second edition). San Francisco: Morgan Kaufmann. 3(1992) 192-213.

[10] Data Hiding in JPEG Images S. K. Muttoo1, Sushil Kumar2 1Reader, Department of Computer Science, University of Delhi, India 2Reader, Rajdhani College, University of Delhi, New Delhi ,India

[11] M. Chen, N. Memon, E.K. Wong, Data hiding in document images, in: H. Nemati (Ed.). Premier Reference Source–Information Security and Ethics: Concepts, Methodologies, Tools and Applications, New York: Information Science Reference, 2008, pp. 438-450.

[12] D.C. Lou, J.L. Liu, H.K. Tso, Evolution of information – hiding technology, in H. Nemati (Ed.), Premier Reference Source–Information Security and Ethics: Concepts, Methodologies, Tools and Applications, New York: Information Science Reference, 2008, pp. 438-450.

[13] Schneider, Secrets & Lies, Indiana:Wiley Publishing, 2000.

[14] E. Cole, Hiding in Plain Sight: Steganography and the Art of Covert Communication, Indianapolis: Wiley Publishing, 2003.

[15] T. Jahnke, J. Seitz, (2008). An introduction in digital watermarking applications, principles and problems, in: H. Nemati (Ed), Premier Reference Source–Information Security and Ethics: Concepts, Methodologies, Tools and Applications, New York: Information Science Reference, 2008, pp. 554-569.