International Journal of Electronics Communication and Computer Engineering
Volume 3, Issue 1, ISSN 2249 –071X

A Secure Image Based Steganographic Model Using RSA Algorithm and LSB Insertion

Swati Tiwari[1], R. P. Mahajan[2]
[1]*Research Scholar,Department Of Computer Science & Engineering, IES, IPS Academy, Indore, swati.tiwari1581@gmail.com*
[2]*Professor, Department Of Computer Science & Engineering, IES, IPS Academy, Indore, rpmahajan@yahoo.com*

Abstract - Steganography is the term used to describe the hiding of data in images to avoid detection by attackers. It is an emerging area which is used for secured data transmission over any public media. In this study a novel approach of image steganography based on LSB (Least Significant Bit) insertion and RSA encryption technique for the lossless jpeg images has been proposed. This paper discusses an application which ranks images in a users library based on their suitability as cover objects for some data. The data is matched to an image; so there is less chance of an attacker being able to use steganalysis to recover the data. The application first encrypts the data using RSA algorithm. The message bits are embedded into the image using Least Significant Bits insertion. Before embedding, the message bits are encrypted using RSA algorithm, resulting in increased robustness. This would decrease the intentional attacks which try to reveal the hidden message. At the receiver side reverse operation has been carried out to get back the original information.

Keywords – Data Hiding, Encryption Technique, Image Steganography, LSB, Steganalysis.

I. INTRODUCTION

Recently, information hiding techniques have attracted much research interests from the field of information security [1]. This paper discusses a data hiding application using steganography. The purpose of this paper is to create a user friendly steganography application that allows users to hide private data in image files. The goal is to make this steganography application less vulnerable to steganalysis.

Steganography is one of the most vital research subjects in the field of security communications. It differs from cryptography in the sense that where cryptography focuses on keeping the contents of a message secret, steganography focuses on keeping the existence of a message secret [2, 3]. Another form of information hiding is digital watermarking, which is the process that embeds data called a watermark, tag or label into a multimedia object such that watermark can be detected or extracted later to make an assertion about the object. The object may be an image, audio, video or text only [4].

Steganography makes the job of the attacker more difficult because the very existence of the asset is hidden. The importance of steganography in maintaining confidentiality can be illustrated with a simple example. Imagine two coworkers, Alisa and Bobb, are communicating with each other over the internet. Jack, an attacker, has access to this communication link, so she eavesdrops on Alisa and Bobb's communications. If Alisa is asking Bobb if he is free for lunch, then Alisa probably does not mind if Jack reads this message. Thus, Alisa can send her query to Bobb along the communication link in plain text. However, if Alisa is sending Bobb confidential information, such as specifications for their company's latest project, then she probably does not want Jack to be able to read these messages. Therefore, Alisa will likely encrypt her messages. A problem arises because the encrypted text is likely garbled, nonsensical data. Thus, Jack, even though she cannot read the encrypted messages, will know that Alisa has a secret that she is sending to Bobb. Jack can then take the encrypted message and attempt to crack it. This is a very real problem because as computational power increases, encryption is becoming easier to break [5]. However, if Alisa uses steganography, and hides her secret message in a generic image file, then she can transmit her secret message to Bobb without evoking Jack's suspicion. For instance, Alisa can hide her secret message in a picture of her garden. She can then send the image, with the secret message hidden inside it, to Bobb. Jack will think Alisa is just sending Bobb a harmless picture, so she will ignore that communication between Alisa and Bobb. Thus, Alisa and Bobb defeat Jack. An assumption can be made based on this model is that if both the sender and receiver share some common secret information then the corresponding steganography protocol is known as secret key steganography where as pure steganography means that there is none prior information shared by sender and receiver. If the public key of the receiver is known to the sender, the steganographic protocol is called public key steganography [6, 7]. For a more thorough knowledge of steganography methodology the reader may see [8, 3]. Although all digital file formats can be used for steganography, but the image and audio files are more suitable because of their high degree of redundancy [3]. Fig. 1 shows the different categories of file formats that can be used for steganography techniques.

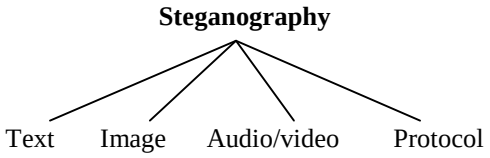


Fig.1: Steganography Category

A block diagram of a generic image steganographic system is given in Fig. 2. A message is embedded in a digital image (cover image) through an embedding algorithm, with the help of a secret key. The resulting stego image is transmitted over a channel to the receiver where it is processed by the extraction algorithm using the same key. During transmission the stego image, it can be monitored by unauthenticated viewers who will only notice the transmission of an image without discovering the existence of the hidden message.

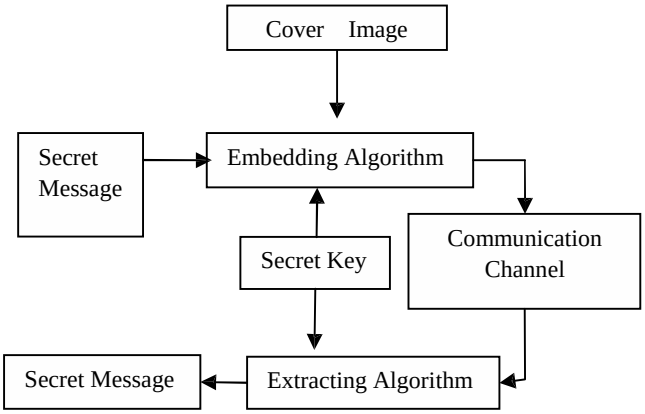


Fig. 2: Image Steganographic System

As mentioned in the example, attackers have more computing power now than ever before. This means that attackers are better able to break encryption algorithms and these capabilities will only increase in the future. DES, an encryption standard that was used by many national governments, successfully withstood attacks for many years since the mid 1970s.However, Biham et al mentions a cryptanalytic attack that can break DES in only a few minutes [9]. Another example of a broken encryption algorithm is WEP. WEP was designed to provide confidentiality to users on wireless networks. Stubbfield et al illustrates how WEP can be broken within hours. DES and WEP are examples of two encryption algorithms that were thought to be secure at the time of their design, but were broken in the future when attackers had more powerful computational resources [10]. These examples prove that encryption is not enough to stop attackers from gaining access to confidential information. It must also be employed to protect confidential assets from being compromised by attackers.

Steganography applications that hide data in images generally use a variation of least significant bit (LSB) embedding [11]. In LSB embedding, the data is hidden in the least significant bit of each byte in the image. The size of each pixel depends on the format of the image and normally ranges from 1 byte to 3 bytes. Each unique numerical pixel value corresponds to a color; thus, an 8-bit pixel is capable of displaying 256 different colors [12].Given two identical images, if the least significant bits of the pixels in one image are changed, then the two images still look identical to the human eye [11]. This is because the human eye is not sensitive enough to notice the difference in color between pixels that are different by 1 unit [11]. Thus, steganography applications use LSB embedding because attackers do not notice anything odd or suspicious about an image if its pixel's least significant bits are modified [11].

Unfortunately for every computer security strategy, there are attackers who develop countermeasures to defeat that security strategy. Steganography is no different; attackers combat steganography using steganalysis. Steganalysis is a process where attackers analyze an image to determine whether it has hidden data in it. A common steganalysis approach is to graph the pixel values of an image that is suspected of containing hidden data. Statistical analysis is then performed on the graphed pixel values [13]. The attackers hope to find anomalies in the statistical analysis of these images. These anomalies may indicate that the image contains a hidden message, and the anomalies may offer some insight into how to extract the hidden message. In this paper a specific image based steganographic model has been proposed which uses a JPEG lossless image as the cover data and the secret information is embedded in the cover to form the stego image. Before embedding the secret information has been encrypted using the public key RSA algorithm. This application also provides a utility to convert JPEG images in the lossless JPEG images. We choose JPG images for this application because most images shared by people today are in the JPG format. Thus to an attacker, the fact that an image other than that of JPEG format is being transferred between two entities could hint of suspicious activity. So this can be an improvement by using JPEG Lossless images.

To combat steganalysis, this application performs an analysis on the user's library of images. This analysis allows users to hide their data in the image that is least likely to be vulnerable to steganalysis.

Section II describes the proposed model. Section III describes different procedures for different processes used at both at sender side and receiver side. Results and discussions are shown in Section IV. Section V describes the implementation part and section VI draws the conclusion.

II. PROPOSED MODEL

Fig. 3 shows the flow chart for the proposed image steganographic model. A user friendly GUI is also incorporated with this application.

Sender Side : The input message is in the text form, and is converted into a bit stream. The application then encrypts this data using the recipient's RSA public key. Each bit of the encrypted data is compared to the least significant bit of

the pixel bytes in an image. The comparisons are made starting from the first byte in the image until the last byte that permits all the data to be hidden in that image. The application cycles through the pixels of the image looking for the block of bytes that result in the least number of LSB changes. The image is then given a rank based on the percentage of least significant bits that match the encrypted data bits. Consider, for example 10 bits of encrypted data that need to be hidden in an image with a bit pattern of 10000000001. If some block of bytes in the image has least significant bits with a pattern 1000000011, this would result in the image receiving a ranking of 90%, because nine of the ten bits are an exact match.

Each image in the user's library is ranked as described above and the user is presented with the list of ranked images. The user is then free to choose which image to use to hide the data. The application does not automatically select the highest ranked image. After hiding data, the stego image is obtained.

Receiver Side : At the receiver side, the data or message is obtained from the stego image. Then this message is decrypted only by the recipient's RSA Private Key.

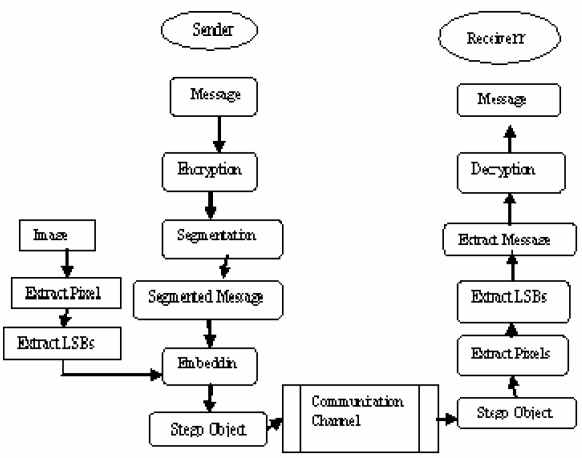
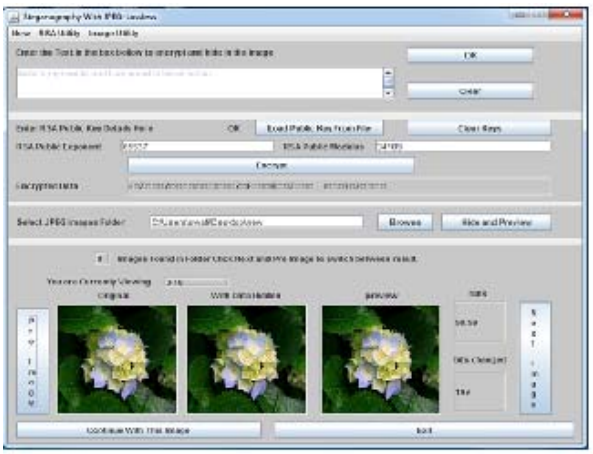
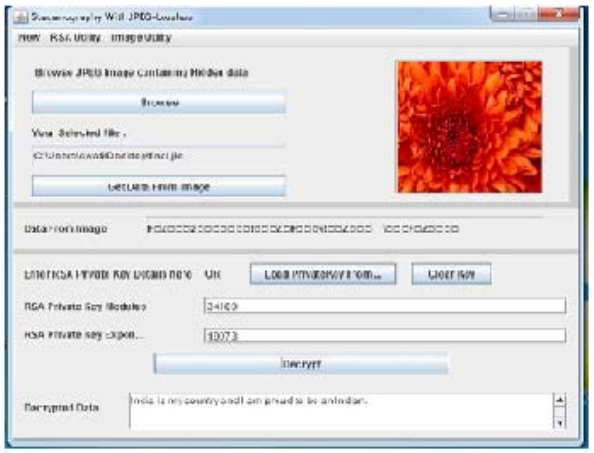


Fig 3: Proposed Steganography System



(a). Sender side



(b). Receiver side

Figure 4: GUI based Steganographic system

III. PROCEDURES FOR PROPOSED SYSTEM

In this section, procedures for different processes used both in the sender side and receiver side are discussed.

Procedures For Hide Text: This procedure hides the encrypted data into the image by searching the best position in the image. The best position defines those Least Significant Bits of the image which extremely match with the encrypted data bits. The output of this procedure is the updated image in which data is hidden.

(a) Procedure **HideText;**
Input : message, publicKeyExponet, publicKeyModulus, imageFolderPath.
Output : finalImage(steganographic image containing message), rank
Start
encryptedData = encrypt(message,publicKeyExponent,publikKeyModulus)
For all image image in imageFolderPath
  Declare pixels[height(image)][width(image)];
  Declare totalchange
  Declare image LSBarray[length(pixels)]
  pixels = getpixel(image);
  ImageLSBarray = getLSBs(Pixels);
  bestPositionList = getBestPositionList(imageLSBarray,encyptedData, totalchange)
Set:
updatedImage = getStegaImage(pixels, encryptedData, bestPositionList)
Rank:=(length(imageLSBarray)-totalchange) / length(imageLSBarray)
For image selected by User
  finalImage:= updateImage ofimage

save bestPositionList to filesystem
save finalImage to filesystem
End;

(b) Procedure
getBestPositionList(imageLSBarray,encyptedData);
Output: PosList pos (List of position elements), totalchange
Start
Totalchange:= 0
For i = 0 to length(encyptedData)-1
Set: index = 0
Set: result = 32
For j = 0 to length(imageLSBarray)-1
If imageLSBarray(j) = -1
Continue Set:
temp=getBitChanges(imageLSBarray(j),encryptedData(i)
If temp < result
Set: Index = b
Set: result = temp
Set: imageLSBarray(index) = -1
Totalchange:=totalchange + result
Set: pos(j) = index
Return pos
End;

(c) Procedure **getStegaImage**(pixels, encData, bestPoss)
Output updatedImage
Start
Set: Pixelslsbarray = getLSBs(pixels);
Set: encBitArray = getBitArray(encData)
For i = 0 to length(bestPoss)-1
For j = 0 to 31
Set: pixelslsbarray(bestPoss(i)*32 + j) = encBitArray(i*32 + j)
Update pixels array by pixelslsbarray
updatedImage := getImageFrompixels(pixels)
Return pos
End;

(d) Procedure **getpixel**(image);
Output: pixles (Two dimensional List of pixel in image)
Start
For i = 0 to height(image)-1
For j = 0 to width(image)-1
Pixels(i)(j) = image.getRBG(j,i)
Return pixels
End;
(e) Procedure **encrypt**(message,e,n)**;**
Output: ciphertext C (List of encrypted int elements)
Start
For i = 0 to length(message)-1
C(i) = message(i)^{e} mod n
Return C
End;

Procedure for Reveal Text: This procedure reveals the secret message which is hidden in the best position of the stego image. This message is in the encrypted form so the message is decrypted by the receiver's private key. Then the original message is presented to the receiver.

(a) Procedure **Reveal Text;**
Input: image, privateKeyExponet, privateKeyModulus,posfile (data position file).
Output: information
Start
encryptedData = getEncData(image,posfile)
information := decrypt(encryptedData,privateKeyExponent, privateKeyModulus)
End;

(b) Procedure **getEncData** (image,posfile)
Output data
Start
Set: pixels = getPixels(image)
Set: encArray = LSBs(pixles)
Set: bestPositions := read data from posfile
For i = 0 to length(bestPositions)-1
data(i) = encArray(bestPosition(i))
return data
End;

(c)Procedure **getpixel**(image);
Output: pixles (Two dimensional List of pixel in image)
Start
For i = 0 to height(image)-1
For j = 0 to width(image)-1
Pixels(i)(j) = image.getRBG(j,i)
Return pixels
End;

(d)Procedure **getLSBs**(Pixels);
Output: LSBs lsb (List of lsb of RGB element in pixels)
Start
For i = 0 to length(Pixels)-1
For j = 0 to 2
lsb(3*I + j) = lsb of byte j in pixel(i)
Return C
End;

(e)Procedure **decrypt**(message,d,n)**;**
Output: plaintext P
Start
For i = 0 to length(message)-1
P(i) = message(i)^{d} mod n
Return P
End;

IV. IMPLEMENTATION

The proposed algorithm is implemented using Java (JDK 1.6). A utility is developed for RSA algorithm. With the help of this utility user can generate public and private keys by providing two prime numbers. The user can enter prime numbers according to his/her choice and an automatic generated prime numbers are also provided as an aid to the user. The utility generates a pair of public and private key, which can be saved in the file or user may remember these keys.

The image utility takes the jpeg image or the folder of images (users library) and convert them into lossless jpeg images. The path is provided by the user in which these converted images are saved.

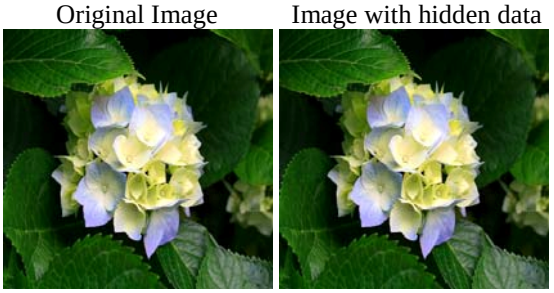
By the "hide text" utility sender can hide his/her secret message which he required to be send to the receiver. In the text box user can type the secret message and this message is encrypted with the receiver's RSA public key. This key is loaded from the file where it was saved. Finally the user has its encrypted form. After this the user's library (the jpeg lossless images) is selected and the encrypted message is hiding in all images of the library or folder. This application determines the rank of the images according to the number of bits changed that is the bits of the encrypted message is compared with the Least Significant bits of the whole image. Where the maximum number of bit match is found the message is hiding in that place. In other words the maximum is the rank less number of bits is changed in the image. This process is repeated for all the images in the user's library. Then the ranking is available for all the images. The user is then free to choose which image to use to hide the data. The application does not automatically select the highest ranked image. The reason this final choice is left to the user is because while an image might be most suited to hiding the data, the image may not be one you would like to share.

The stego image is obtained by user at the receiver side. The user then gets the message in the encrypted form. The recipient then decrypts the message only with his private key which is made available to him through any media. After decryption user gets the original message. The reveal text utility is developed for this.

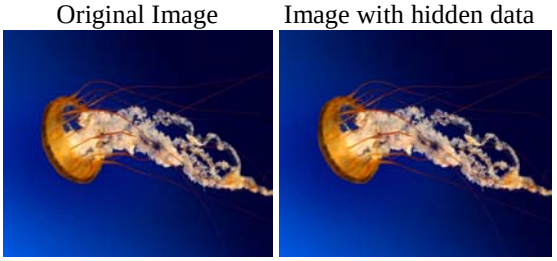
This application is tested on a set of following six images for which different ranking is determined by the program. This study is proceeding with the 'Chrysanthemum' image because it was ranked at 99.93% by this utility.

(a) Desert- Original Image Image with hidden data

(b) Chrysanthemum-
Original Image Image with hidden data

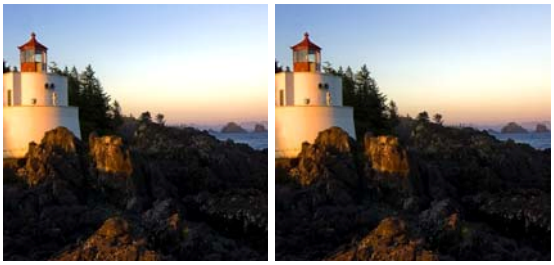


(c) Hydrangeas –
Original Image Image with hidden data



(d) Jellyfish - Original Image Image with hidden data

(e) koala - Original Image Image with hidden data



(f) Lighthouse - Original Image Image with hidden data

Figure 5: original image and image with the hidden data

The following table summarizes the ranking provided by this application:

S.No.	Image	Rank(in percentage)
1.	Desert	98.33
2.	Chrysanthemum	99.93
3.	Hydrangeas	99.12
4.	Jellyfish	98.21
5.	Koala	99.11
6.	Lighthouse	98.23

Table 1: Images with the ranking provided by the application

V. RESULTS AND DISCUSSIONS

In order to test the proposed technique, the 'Chrysanthemum' image of size 1024 x 768 is used. The selected images are converted to lossless JPEG format. The cover image used to hide data is shown in Figure 6(a). The data hidden in the file is the string 'India is my country and I feel proud to be an Indian' encrypted using a public key. The image with the hidden data is shown in Figure 6(b). These images are visually identical as can be seen.

Fig.6: (a). Original Image (b). Image with hidden data

A graph between the number of characters in the message and the bits changed in the image is plotted. The graph is shown in the fig. 7. It is observed that the numbers of hidden characters are proportional to the numbers of LSBs changed in the image i. e. the graph grows linearly. It shows that the number of bits changed in the image with respect to the number of characters in the message. So the distortion in the image will be less.

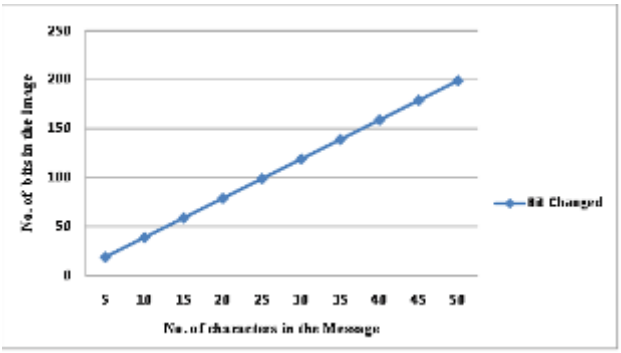


Fig 7. Show the number of characters goes on increase- the number of bits in the message also increases.

Fig. 8(a) shows the portion of the same image. The portion of the above image of fig. 6(b) is magnified three times and fig. 8(b) shows the corresponding image equally magnified following steganography with this application. It can be seen that there is no visually difference between the two images. This in itself does not necessarily alert the existence of steganography to an attacker.

Fig. 8(a) Original Image - Magnification x3
(b) Image with hidden data -

Fig. 8(b) Image with hidden data - Magnification x3

The previous work has done with the bmp file format. BMP files are bigger compared to other formats which render them improper for network transmissions and to hide a secret message inside a BMP file, one would require a very large cover image and moreover most images shared by people today are in the JPG format. The act of sending other images in itself could cause an attacker to be suspicious of the image. This establishes the fact that this approach of steganography model has improved level of security by incorporating the idea of RSA algorithm, with the use of encrypted form of the original message. The use of private key by the receiver to decrypt the message has further enhanced the level of security and make sure that the receiver is the intended recipient of the message.

VI. CONCLUSION

A new and efficient steganographic method for embedding secret messages into images without producing any major changes has been proposed here through LSB method. In this paper authors have used the two stage security for message hiding and transmission. A combination of LSB insertion and the RSA public key algorithm is used to obtain secure stego image. The RSA algorithm has been used to generate the encrypted form of the message in order to achieve high level of security. The encrypted form of the message is embedded into the cover image. This property enables the method to avoid steganalysis. The integrated approach of combining the above two methods enable secure transfer of the message compared to earlier techniques. Presently, this application supports hiding data in lossless jpg images. Future improvement of this application would be extending its functionality to support hiding data in video files or in other file format.

REFERENCES

[1] Tao Zhang, Wenxiang Li, Yan Zhang, Xijian Ping" Detection of LSB Matching Steganography Based on Distribution of Pixel Differences in Natural Images" , Publication Year: April-2010, Page(s): 548 – 552
[2] Ross J. Anderson and Fabien A.P. Petitcolas, "On the limits of steganography," IEEE Journal on Selected Areas in Communications (JSAC), Special Issue on Copyright & Privacy Protection, Vol. 16 no. 4, pp 474-481, May 1998.
[3] T Mrkel, JHP Eloff and MS Olivier ."An Overview of Image Steganography,"in proceedings of the fifth annual Information Security South Africa Conference, 2005.
[4] Digital Watermarking: A Tutorial Review S.P.Mohanty, 1999.
[5] J. Siegfried, C. Siedsma, B.J. Countryman, C.D. Hosmer, "Examining the Encryption Threat," International Journal of Digital Evidence, Vol. 6, pp. 23-30, December 2004.
[6] "Stretching the Limits of Steganography", RJ Anderson, in Information Hiding, Springer Lecture Notes in Computer Science v 1174 (1996) pp 39-48
[7] Scott Craver, "On Public-key Steganography in the Presence of an Active Warden," in Proceedings of 2nd International Workshop on Information Hiding, April 1998, Portland, Oregon, USA. pp. 355 - 368.
[8] N. F. Johnson and S. Jajodia, "Steganography: seeing the unseen," IEEE Computer., Feb., 26-34 (1998).
[9] E. Biham, A. Shamir. "Differential cryptanalysis of DES-like cryptosystems," Journal of Cryptology, Vol. 4, pp. 3-72, January 1991
[10] A. Stubblefield, J. Ioannidis, A. D. Rubin, "A Key Recovery Attack on the 802.11b Wired Equivalent Privacy Protocol (WEP)," ACM Transactions on Information and System Security, Vol. 7, pp. 319-332, May 2004.
[11] R. Chandramouli, N. Memon, "Analysis of LSB based image steganography techniques," Image Processing, Vol. 3, pp. 1019-1022, October 2001.
[12] J. D. Foley, Computer Graphics: Principles and Practicies. Cornell: Addison-Wesley, 1996, pp. 3..
[13] J. Fridrich, M. Long, "Steganalysis of LSB encoding in color images,"Multimedia and Expo, vol. 3, pp. 1279-1282, July 2000.
[14] Crytography & Network security: Principles and Practices by William Stalling.

AUTHOR'S PROFILE

Prof. R. P. Mahajan has completed his graduation from SGSITS, Indore, India in 1972 and his M.E. From Indian Institute of Science, Banglore, India in 1974. He received M. Tech from Birla Institute of Technology, Misra, and Ranchi, India.

He served IT industry and academics for more than 35 years in various positions. Currently he is professor in the department of Computer Science & Engineering at Institute of Engineering and Science, IPS Academy, Indore, India. He has published 4 papers in international societies or journals. Presently he is pursuing his Ph.D. from DAVV, Indore, India. He is a life member of Computer Society of India and Institute of Engineers. He is also a member of Institute of Electrical and Electronics Engineers. His research interests include data mining, neural networks, and database management system and quantum computation.

Swati Tiwari received B.E. degree in Information Technology from SGSITS, Indore, India in 2004 and she is a research scholar at IES, IPS Academy. She is a member of Computer Society of India and has presented research articles in national conferences.