A FAST PARTIAL IMAGE ENCRYPTION SCHEME WITH WAVELET TRANSFORM AND RC4

Sapna Sasidharan and Deepu Sleeba Philip
Software Engineer, iGATE Patni Global Solutions, Chennai, India.

ABSTRACT

Encryption is used to securely transmit data in open networks. Each type of data has its own features; therefore different techniques should be used to protect confidential image data from unauthorized access. In this paper, a fast partial image encryption scheme using Discrete Wavelet Transform with RC4 Stream Cipher is done. In this method, the approximation matrix (lowest frequency band) is encrypted using the stream cipher as it holds most of the image's information. The encryption time is reduced by encrypting only the part of the image and maintains a high level of security by shuffling the rest of the image using the shuffling algorithm. Selective encryption is a recent approach to reduce the computational requirements for huge volumes of images.

KEYWORDS: *DWT, Stream Cipher, Shuffling Algorithm, Selective Encryption*

I. INTRODUCTION

The field of encryption is becoming very important in the present era in which information security is of utmost concern. Security is an important issue in communication and storage of images, and encryption is one of the ways to ensure security. Image encryption has applications in internet communication, multimedia systems, medical imaging, telemedicine, military communication, etc. Information security is becoming more important in data storage and transmission. Images are widely used in several processes. Therefore, the protection of image data from unauthorized access is important. Image encryption plays a significant role in the field of information hiding [1].

There are two basic ways to encrypt digital images: in the spatial domain or in the transform domain [2]. Since wavelet based compression appeared and was adopted in the JPEG2000 standard, suggestions for image encryption techniques based in the wavelet domain have been abundant. However, many of these are not secure as they are based exclusively on random permutations making them vulnerable to known or chosen-plaintext attacks [2]–[4]. The encryption scheme presented here is based on the DWT and RC4 Stream Cipher. The scheme aims at reducing encryption time by only encrypting part of the image, yet maintaining a high level of security by shuffling the rest of the image using the Shuffling Algorithm.

The idea here is to encrypt the approximation matrix (*ca*) with the stream cipher as it holds most of the image's information**. S**tream ciphers typically encrypt one byte at a time. To generate a stream cipher a key is input into a random number generator. The generator produces a keystream consisting of random numbers, each 8 bits long. For a high level of security, the keystream should be unpredictable without knowledge of the input key. The keystream is combined with the plaintext using the bitwise exclusive**-**OR (XOR). In symmetric encryption, the same key is used for encryption and decryption [5], [6]. While encrypting this matrix alone will provide complete perceptual encryption, it would be possible for an attacker to gain information about the image from the other matrices, especially in images that have a lot of edges. Therefore, the horizontal (*ch*), vertical (*cv*), and diagonal (*cd*) matrices will be shuffled using the Shuffling Algorithm.

II. DISCRETE WAVELET TRANSFORM

Wavelets are mathematical functions that cut up data into different frequency components. Wavelet algorithms process data at different scales or resolutions. The wavelet transform carries out a special form of analysis by shifting the original signal from the time domain into the time–frequency, or, in this context, time–scale domain. It is illustrated in Figure 1. The idea behind the wavelet transform is the definition of a set of basis functions that allow an efficient, informative and useful representation of signals.

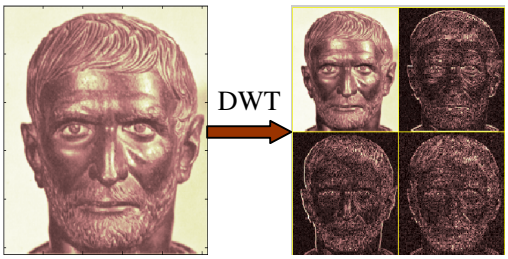


Figure 1. DWT Illustration

A wavelet is a function $\psi \varepsilon L_2$ (R) which meets the admissibility condition is written in equation

$$0 < C_\psi := 2\pi \int_R \frac{\varphi(w)2}{w} \, d\omega < \infty \qquad (1)$$

where, ψ denotes the Fourier transform of the wavelet ψ.
The constant C_ψ designates the admissibility constant ω denotes the signal to be transformed. Approaching $\omega \longrightarrow 0$ gets critical. To guarantee that the above equation (1) is accomplished, we must ensure that $\psi(0) = 0$.
Since $\psi \varepsilon L_2$ (R), also is its Fourier transform $\psi \varepsilon L_2$ (R): $\int_R \varphi(\omega)2 \; d\omega < \infty$.
Therefore, $\varphi(\omega)$ declines sufficiently fast for $\omega >> 0$. In practical considerations, it is sufficient that the majority of the wavelet's energy is restricted to a finite interval. This means that a wavelet has strong localization in the time domain.

2.1 Daubechies Wavelet

The family of Daubechies wavelets is most often used for multimedia implementations. They are a specific occurrence of the conjugate-quadrature filters. The Daubechies wavelets (see Figure 2) are obtained by iteration; no closed representation exists. The Daubechies wavelets are the shortest compactly supported orthogonal wavelets for a given number of vanishing moments. The degree n_0 of vanishing moments determines the amount of filter bank coefficients to $2n_0$. After embedding, the stego-image will be inverse transformed to the spatial domain. The inverse transforms (IDWT) takes the values of the frequency domain and transfers them back into the time domain.

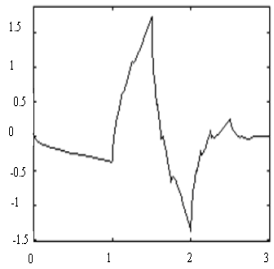


Figure 2. Daubechies Wavelet

III. STREAM CIPHER

RC4 is a stream cipher, symmetric key algorithm. The same algorithm is used for both encryption and decryption as the data stream is simply XORed with the generated key sequence. The keystream is completely independent of the plaintext used. It uses a variable length key from 1 to 256 bit to initialize a 256-bit state table. The state table is used for subsequent generation of pseudo-random bits and then to generate a pseudo-random stream which is XORed with the plaintext to give the ciphertext.

The algorithm can be broken into two stages: initialization, and operation. In the initialization stage the 256-bit state table, **S** is populated, using the key, K as a seed. Once the state table is setup, it continues to be modified in a regular pattern as data is encrypted.

The initialization process can be summarized by the pseudo-code:

```
j = 0;
for i = 0 to 255:
S[i] = i;
for i = 0 to 255:
j = (j + S[i] + K[i]) mod 256;
swap S[i] and S[j];
```

It is important to notice here the swapping of the locations of the numbers 0 to 255 (each of which occurs only once) in the state table. The values of the state table are provided. Once the initialization process is completed, the operation process may be summarized as shown by the pseudo code below;

```
i = j = 0;
for (k = 0 to N-1) {
i = (i + 1) mod 256;
j = (j + S[i]) mod 256;
swap S[i] and S[j];
pr = S[ (S[i] + S[j]) mod 256]
output M[k] XOR pr
}
```

where, M[0..N−1] is the input message consisting of N bits.

This algorithm produces a stream of pseudo-random values. The input stream is XORed with these values, bit by bit. The encryption and decryption process is the same as the data stream is simply XORed with the generated key sequence.

Some of the RC4 algorithm features can be summarized as:

1. Symmetric stream cipher
2. Variable key length
3. Very quick in software
4. Used for secured communications as in the encryption of traffic to and from secure web sites using the SSL protocol.

IV. PROPOSED METHOD

In the DWT method, the image first goes through the single-level DWT resulting in four coefficient matrices; the approximation (*ca*), horizontal (*ch*), vertical (*cv*), and diagonal (*cd*) matrices. The lowest frequency sub-band is expressed in the matrix *ca*. The *ca* matrix will be encrypted as it holds most of the image's information using the RC4 Stream Cipher. For encryption, the RC4 keystream will be combined with the *ca* coefficients using the XOR operation. While encrypting this matrix alone will provide complete perceptual encryption, it would be possible for an attacker to gain information about the image from the other matrices. Therefore, the horizontal (*ch*), vertical (*cv*), and diagonal (*cd*) matrices will be shuffled. The Shuffling Algorithm used in the DCT [7] [8] method is used here. The encrypted *ca* matrix and the shuffled *ch*, *cv* and *cd* matrices then undergo the Inverse Discrete Wavelet Transform (IDWT) to produce the encrypted image. This method aims at reducing encryption time by only encrypting part of the image, yet maintaining a high level of security by shuffling the rest of the image. Figure 3 shows the block diagram of the proposed system.

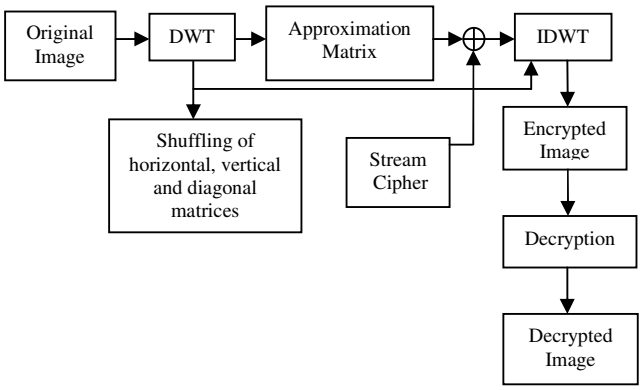


Figure 3. Block Diagram of the Proposed System

In the following, the encryption, decryption and shuffling of the images are illustrated.

Algorithm to Encrypt Image

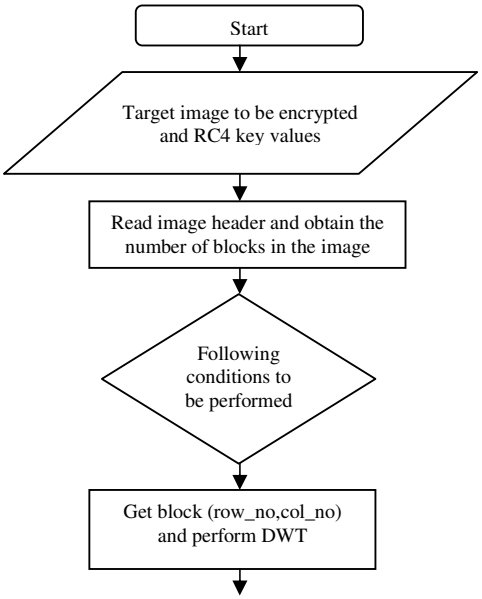
Input : Target Image to be encrypted and the stream RC4 Key values.
Output : Encrypted Image
Begin
Step 1: Read the image header, save the height of the image in variable height & the width in variable width and save the body image in an array imagebody.
Step 2: Obtain how many blocks exist in an image row and how many ones in the column, by dividing the width and height of the image by N, where N is equal to 8 (the required block size).

- NoRowB = Image Height / N;
- NoColB = Image Width / N;

Step 3: For all blocks in the image perform the following:

- Get_block (row_no, col_no)
- Perform a DWT on the block and save the resulted coefficients in an array.
- Round the selected coefficients, convert the selected coefficients to 11 bits.
- Encrypt the selected coefficients by XORing the generated bit stream from the RC4 + Key with the coefficient bits, the sign bit of the selected coefficients will not be encrypted.
- Perform an Inverse Discrete Wavelet Transform (IDWT) and get the new block values and the resulted values could be positive or negative values due to the encryption step.

Step 4: Apply the proposed shuffling algorithm on the resulted blocks to obtain the encrypted image.
End
Various steps for encrypting the image is shown in figure 4.



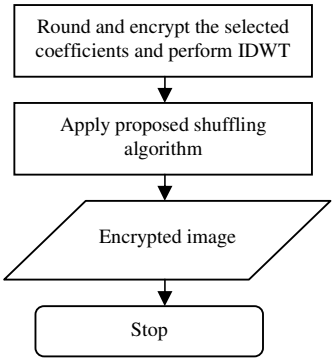


Figure 4. Flowchart for encrypting the image

Algorithm to Decrypt Image

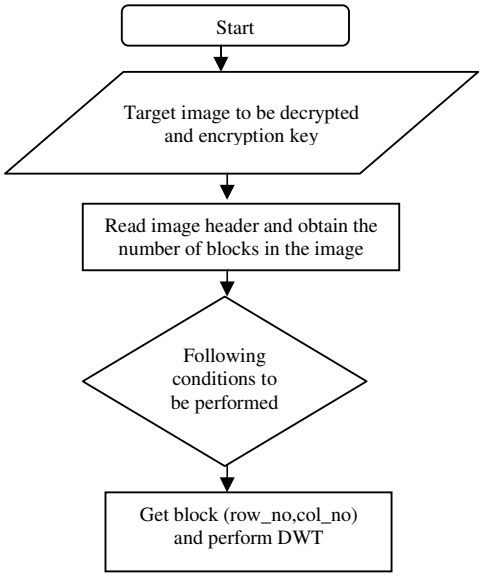
Input : Target Image to be decrypted and the Encryption Key
Output : Original Image
Begin
Step 1: Read the image header, save the height of the image in variable height & the width in variable width and save the body image in an array imagebody.
Step 2: Obtain how many blocks exist in an image row and how many ones in the column, by dividing the width and height of the image by N, where N is equal to 8 (the required block size).

- NoRowB = Image Height / N;
- NoColB = Image Width / N;

Step 3: For all blocks in the image perform the following:

- Get_block (row_no, col_no)
- Perform a DWT on the block and save the resulted values in an array.
- Round the selected coefficients, convert the selected coefficients to 11 bits.
- Decrypt the resulted bits by using the generated bit stream from the RC4 + Key, by performing an XOR operation, the sign bit of the selected coefficients will remain.
- Convert the resulted bits into integer values, and join the sign (from the step above) with each integer, if the coefficient is negative multiply it by −1.
- Perform an Inverse Discrete Wavelet Transform (IDWT) and get the new blocks.

Step 4: Reshuffle the block, since the shuffling algorithm generates the same row and column numbers to return the shuffled blocks into their original locations.
Step 5: Reconstruct the image to get the original Image.
End
Various steps for decrypting the image is shown in figure 5.



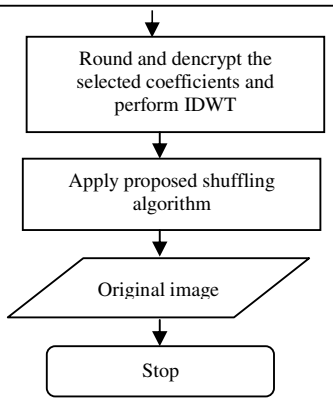


Figure 5. Flowchart for decrypting the image

Shuffling Algorithm

Input : Key, number of blocks in the row (**NoRows**), number of blocks in the column (**NoCols**) and the resulted encrypted image saved in an array.
Output: A new shuffled image
Begin

```
for i = 0 to (NoRows×NoCols)
NewVal[i]=(K×i)mod(NoRows×NoCols)
endfor
K= 0
for i = 0 to (NoRows×NoCols)
MoveBlock(ImageBlk(NewVal[I]), ImageBlk [K])
K++
endfor
```

End
Various steps of shuffling is shown in figure 6.

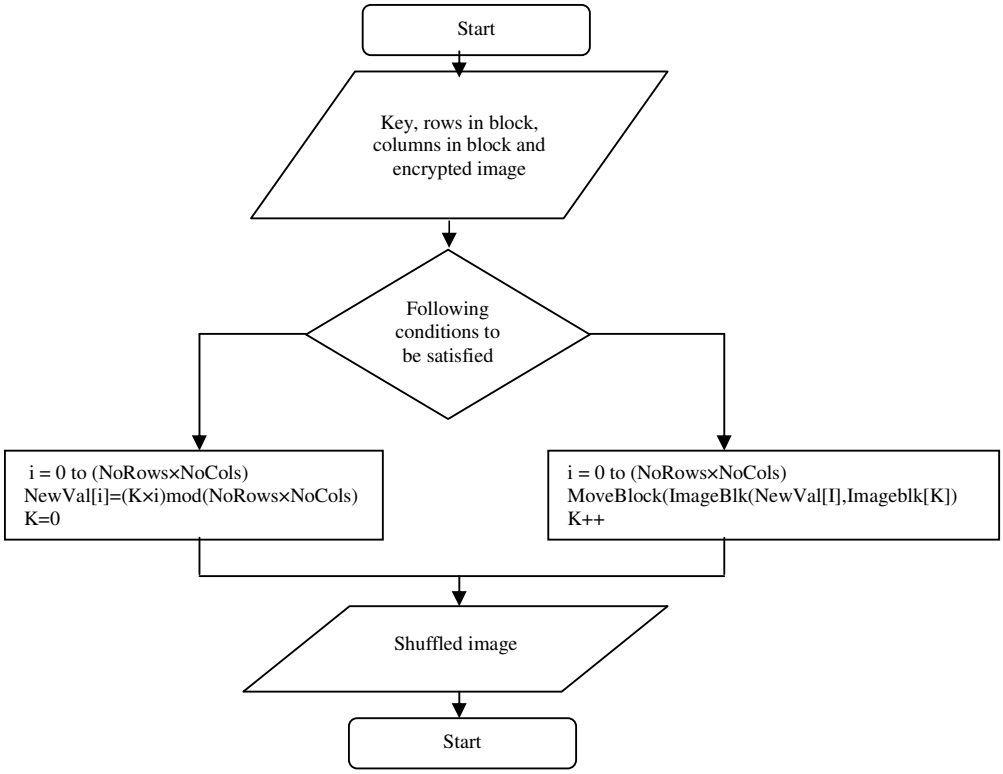


Figure 6. Flowchart of shuffling algorithm

V. EXPERIMENTAL RESULTS

The performance analysis of selective image encryption DWT with Stream Cipher is measured using the Peak Signal to Noise Ratio (PSNR), Histogram Analysis and Entropy. Figure 7 shows the Original Image used in the DWT method [9] [10]. Figure 8 shows the Selective Encryption of the original image. The Encrypted Image after applying the shuffling algorithm is shown in Figure 9 and in Figure 10, the Decrypted Image is shown.

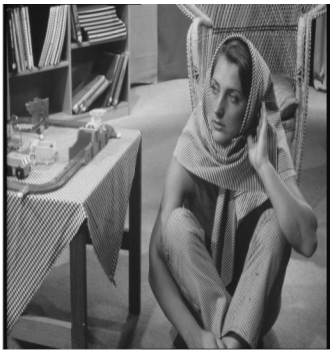


Figure 7. Original Image

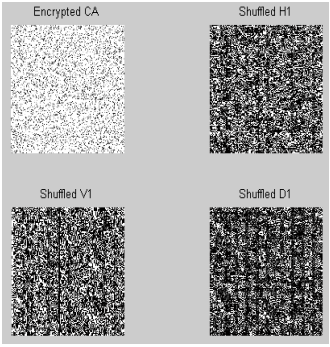


Figure 8. Selective Encryption

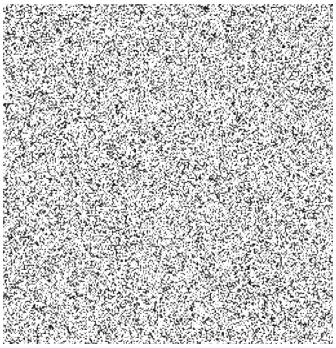


Figure 9. Encrypted Image

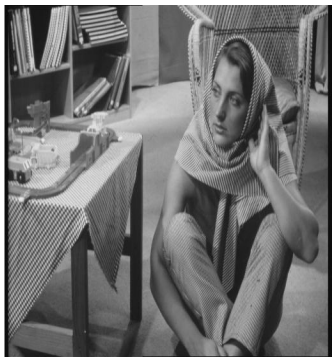


Figure 10. Decrypted Image

Table 1 shows the Performance Analysis of the encrypted and decrypted images in terms of PSNR when tested with different test images of size 512×512. A lower PSNR is obtained in the case of Encrypted Image and a higher PSNR is obtained in the case of Decrypted Image. Higher PSNR value shows a better quality of the image.

Table 1. Performance Analysis of DWT Method

Test Images	**PSNR of Encrypted Image**	**PSNR of Decrypted Image**
Barbara	20.5784	85.6641
House	20.7056	85.4996
Lena	20.8768	85.5393
Airplane	20.6219	85.4215
Baboon	20.7354	85.3072

To demonstrate that our proposed algorithm has strong resistance to statistical attacks, test is carried out on the histogram of enciphered image. Several gray-scale images of size 512×512 are selected for this purpose and their histograms are compared with their corresponding ciphered image. One typical example is shown below. The histogram of the original image contains large spikes as shown in Figure 11 but the histogram of the cipher image as shown in Figure 12, is more uniform. It is clear that the histogram of the encrypted image is, significantly different from the respective histogram of the original image and bears no statistical resemblance to the plain image. Hence statistical attack on the proposed image encryption procedure is difficult.

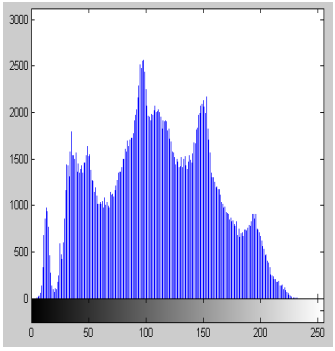


Figure 11. Histogram of Original Image

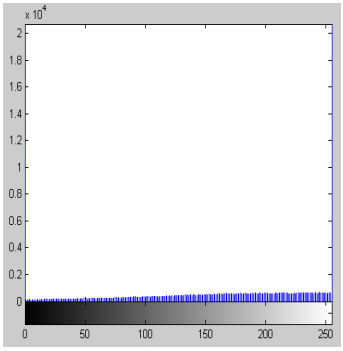


Figure 12. Histogram of Encrypted Image (after shuffling)

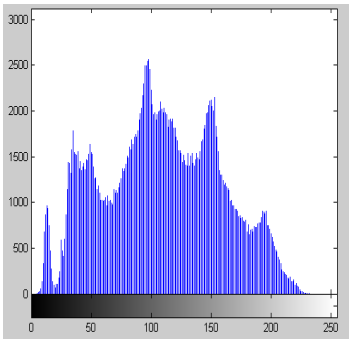


Figure 13. Histogram of Decrypted Image

Entropy is a statistical measure of randomness. Table 2 shows the entropy of different test images of size 512×512.

Table 2. Entropy of different test images

Test Images	**Entropy of Encrypted Image**
Barbara	4.7879
House	4.7888
Lena	4.7807
Airplane	4.7899
Baboon	4.7916

VI. CONCLUSION

A fast partial image encryption scheme for images using DWT with RC4 Stream Cipher has been presented in this paper. The system only encrypts the lowest frequency band of the image, however it is highly secure as the rest of the other bands are all shuffled using the Shuffling Algorithm. The algorithm is considered as a fast image encryption algorithm, due to the selective encryption of certain portion of the image (lowest frequency band). PSNR values of the encrypted images are low and are resistant to statistical attacks. Hence, better security has been provided.

REFERENCES

[1] Said E. El-Khamy, Mohammad Abou El-Nasr, Amina H. El-Zein, *"A Partial Image Encryption Scheme Based on the DWT and ELKNZ Chaotic Stream Cipher",* MASAUM Journal of Basic and Applied Sciences, Vol. 1, No. 3, October 2009.

[2] S. Li, G. Chen, *"Chaos-Based Encryption for Digital Images and Videos"*, in Multimedia Security Handbook, B. Furht and D. Kirovski, CRC Press, 2004.

[3] S. Lian, Z. Wang, *"Comparison of Several Wavelet Coefficients Confusion Methods Applied in Multimedia Encryption"*, In Proc. Int. Conference on Computer Networks and Mobile Computing (ICCNMC'2003), pp. 372–376, 2003.

[4] G. Ginesu, T. Onali, D.D. Giusto, *"Efficient Scrambling of Wavelet-based Compressed Images*: *A comparison between simple techniques for mobile applications"*, Proceedings of the 2nd International Mobile Multimedia Communications Conference (*MobiMedia'06*), 2006.

[5] W. Stallings, *Cryptography and Network Security*, Prentice Hall, New Jersey, 2006.

[6] S. El-Khamy, M. Lotfy, and A. Ali, *"The FBG Stream Cipher,"* Proc. of *URSI-NRSC*, 2007, pp. 1-8.

[7] C. Coconu, V. Stoica, F. Ionescu, D. Profeta, "*Distributed Implementation of Discrete Cosine Transform Algorithm on a Network of Workstations*", Proceedings of the International Workshop Trends & Recent Achievements in IT, Romania, pp. 116-121, May 2002.

[8] Ramazan Gencay, Faruk Selcuk, Brandon Whitcher, "*An Introduction to Wavelets and Other Filtering Methods in Finance and Economics*", Academic Press, 2001.

[9] Lala Krikor, Sami Baba, Thawar Arif, Zyad Shaaban, "*Image Encryption Using DCT and Stream Cipher*", European Journal of Scientific Research, Vol.32, No.1, pp.47-57, 2009.

[10] M. Van Droogenbroeck, R. Benedett, "*Techniques for a Selective Encryption of Uncompressed and Compressed Images*", in Proceedings of Advanced Concepts for Intelligent Vision Systems (ACIVS) 2002, Ghent, Belgium, September 2002.

Authors

Sapna Sasidharan received her B.Tech degree in Computer Science and Engineering from Sree Narayana Guru College of Engineering and Technology, Kannur University, Kerala in 2008. She has completed her M.Tech degree in Cyber Security from Amrita Vishwa Vidyapeetham University, Coimbatore in 2010. Her research interests are Image Encryption, Steganography and Cryptography. She is currently working as a Software Engineer in iGATE Patni Global Solutions, Chennai. She has published 2 papers in International Journals and 3 papers in International Conferences.

Deepu Sleeba Philip received his B.Tech degree in Electronics and Communication Engineering from College of Engineering, Kidangoor, Cusat University, Kerala in 2010. His research interests are Image Encryption and Cryptography. He is currently working as a Software Engineer in iGATE Patni Global Solutions, Chennai.