# Pixel Value Differencing Based Image Steganography using AES and SHA-2 Cryptography Method

**Jayeeta Majumder, Chittaranjan Pradhan**

***Abstract:** The growing use of Internet and availability of public and private digital data and its sharing among professionals and researchers need particular attention to information security. This information needs to be protected against unauthorized access and attacks. Digital communication witnesses a noticeable and continuous development in many applications in the Internet. Hence, secure communication sessions must be provided. The security of data transmitted across a global network has turned into a key factor on the network performance measures. So, the confidentiality and the integrity of data are needed to prevent eavesdroppers from accessing and using transmitted data. Presently, three main methods of information security being used: watermarking, cryptography and steganography. The aim of this paper is to develop a new approach to hiding a secret information in an image by taking advantage of benefits of combining cryptography and steganography. In this method first, the message is encrypted by using AES algorithm and hashed the key using SHA-2 to prevent from attacks. After that, we performed some modifications on PVD algorithm by adding a key to make hiding process non sequential. Results achieved indicate that our proposed method is encouraging in terms of robustness and security.*

***Keywords:** Steganography, AES algorithm, SHA-2, Pixel Value Differencing Method and Data hiding*

## I. INTRODUCTION

Internet use in modern days are increasing rapidly. So, data security is the major issue. In terms of data security, cryptography and steganography both are using to securing the information [1]. For data encryption and decryption the cryptography techniques are used and for data hiding steganography techniques are used. Encryption techniques are used to convert the plain text into cipher text. The decryption techniques are used to convert the cipher text into plain text. At the sender side the encryption process is done and decryption procedure is performed at the receiver side. Cryptography is classified into symmetric cryptography and asymmetric cryptography. The same secret key is used in Symmetric key method for data encryption and decryption method at the sender and receiver end.

The different secret key known as public key and the private key is mainly used for data encryption and decryption in both the sender and receiver side. The Hash function in cryptography method [4] which produces the hash value. For data authentication and data integrity the arbitrary length string known as hash value is used. The hash value is a one way function. The AES algorithm [2,3] is a block cipher cryptography method, with a fixed data block size length of 128 bits with 128, 192, and 256 bits different key lengths to encrypt and decrypt the data bits in a block. The 128 data bits mean that the size of the block for AES operation is a 4x4 square of bytes. [2]. The PVD method is used for steganography techniques. [7, 8].

Depending on the key size the AES parameters are shown in Table (1) below.

**Table 1: The AES Parameters**

| Key size Nk (bits) | Plaintext block size (bits) | Number of rounds Nr (bits) | Round key size (bits) | Expanded key size (bits) |
|---|---|---|---|---|
| 128 | 128 | 10 | 128 | 176 |
| 192 | 128 | 12 | 128 | 208 |
| 256 | 128 | 14 | 128 | 240 |

## II. THE AES ALGORITHM

The Advanced Encryption Standard (AES) method is the most popular and widely used symmetric key encryption technique for cryptography. In comparison to Fiestel cipher AES is an iterative method. The basis of AES method is “Substitution- Permutation Network”. The AES encryption algorithm consist of four byte-oriented different transformation steps. The steps are SubByte or Byte substitution, Shift Rows, MixColumns and AddRoundKey [2].

In AES decryption method same as AES encryption consist of four byte-oriented different transformation steps. The steps are: Inverse SubByte or Inverse Byte Substitution, Inverse ShiftRows, Inverse MixColumns and AddRoundKey [2] as shown in figure (1).

Manuscript published on January 30, 2020.
* Correspondence Author

**Jayeeta Majumder***, Assistant Professor, Dept of Computer Science & Engineering, Haldia Institute of Technology, Haldia, West Bengal, Email: jem2003_kolkata@yahoo.co.in

**Chittaranjan Pradhan**, Associate Professor, Dept. of Computer Science & Engineering, KIIT University, Bhubaneswar, Odissa,

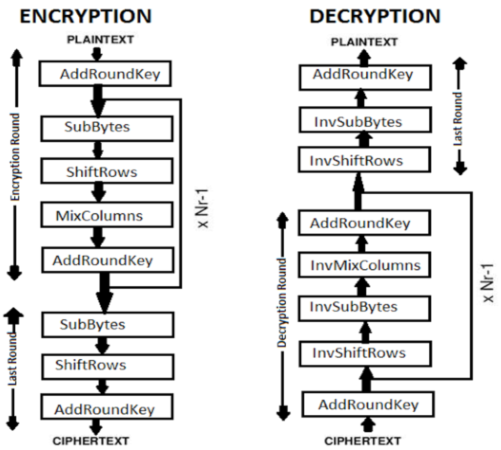


**Fig 1. The AES Encryption and Decryption round**

### A. The SubBytes or Byte Substitution and Inverse SubBytes Procedure

In the AES encryption process, the first step is Substitution Bytes. Here one Substitution Box or S-Box is used in the algorithm, the input data is split into block of bytes and matched with the S-Box. The same S-Box is used for all the data bytes in the AES algorithm. The S-box is mainly use the inverse multiplication technique in the Galois Field $2^8$. The substitutions on an S-box is an invertible nonlinear transformation that works on 8 bits at a time. It uses the following reducing polynomial for multiplication: $x^8 + x^4 + x^3 + x +$ 1. In the AES decryption process, the Inverse Substitution Bytes process are used for every byte of the cipher text data. The encrypted data is matched with the corresponding Inverse SubByte value [2].

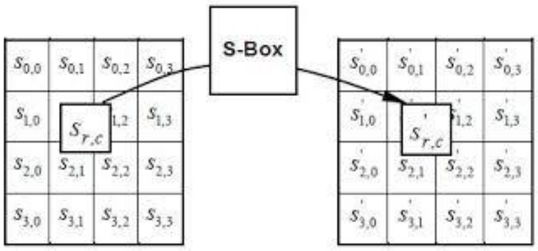

$S_{0,0}$ $S_{0,1}$ $S_{0,2}$ $S_{0,3}$
$S_{1,0}$ $S_{r,c}$ $S_{1,3}$
$S_{2,0}$ $S_{2,1}$ $S_{2,2}$ $S_{2,3}$
$S_{3,0}$ $S_{3,1}$ $S_{3,2}$ $S_{3,3}$
S-Box
$S_{0,0}$ $S_{0,1}$ $S_{0,2}$ $S_{0,3}$
$S_{1,0}$ $S_{r,c}$ $S_{1,3}$
$S_{2,0}$ $S_{2,1}$ $S_{2,2}$ $S_{2,3}$
$S_{3,0}$ $S_{3,1}$ $S_{3,2}$ $S_{3,3}$

**Fig 2. Substitution Bytes phases of AES**

### B. The ShiftRows Transformation and Inverse Shift Row Transformation

In the second step of AES encryption algorithm, it is the shift rows transformation. The row transformation procedure is stated below:

The Shift function generally shifts each bytes in every row of a matrix by using a fixed offset, determined by the encryption algorithm. The rows in the matrix refers to the representation of the internal state in AES. Here data is a 4x4 matrix and each cell contains a byte. The Bytes are present in the matrix across rows from left to right and down columns.

For AES, the first row of the data matrix is remain unchanged. Each byte in the second row is shifted left by one position. Bytes in the third and fourth rows are shifted by two and three position, respectively by the offset value.

For AES decryption process, the Inverse Shift Rows transformation is followed.

- The Inverse ShiftRows step performs these circular shifts in right to left direction for each of the last three rows. Here also the first row remains unchanged. [4, 5]

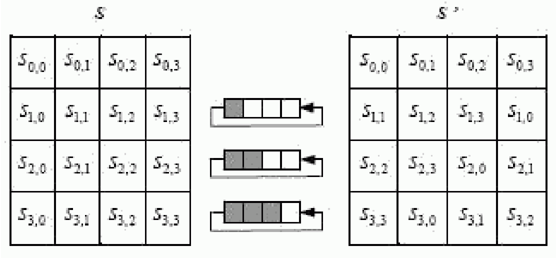

S
S '
$S_{0,0}$ $S_{0,1}$ $S_{0,2}$ $S_{0,3}$
$S_{1,0}$ $S_{1,1}$ $S_{1,2}$ $S_{1,3}$
$S_{2,0}$ $S_{2,1}$ $S_{2,2}$ $S_{2,3}$
$S_{3,0}$ $S_{3,1}$ $S_{3,2}$ $S_{3,3}$
$S_{0,0}$ $S_{0,1}$ $S_{0,2}$ $S_{0,3}$
$S_{1,1}$ $S_{1,2}$ $S_{1,3}$ $S_{1,0}$
$S_{2,2}$ $S_{2,3}$ $S_{2,0}$ $S_{2,1}$
$S_{3,3}$ $S_{3,0}$ $S_{3,1}$ $S_{3,2}$

**Fig 3. Shift Rows Transformation phases of AES**

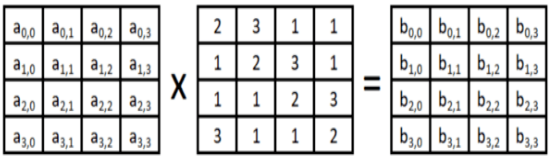
### C. The MixColumn Transformation and Inverse MixColumn Transformation

In the third steps of AES encryption algorithm, the mix column technique is used. MixColumns steps perform operations by splitting the matrix into columns with the use of the prime poly m(x) $=x^8+x^4+x^3+x+1$. Each byte gets a new generated value is the function of all four data bytes in the column. The product matrix is the product of one row and one column element. It also use the arithmetic of GF $(2^{8)}$. Each column operates separately. The MixColumns operation of each column j $(0 \leq j \leq 3)$ is as follows,

$$S'_{0,j} = (2 \bullet c_{0,j}) \oplus (3 \bullet c_{1,j}) \oplus c_{2,j} \oplus c_{3,j}$$
$$S'_{1,j} = c_{0,j} \oplus (2 \bullet c_{1,j}) \oplus (3 \bullet c_{2,j}) \oplus c_{3,j}$$
$$S'_{2,j} = c_{0,j} \oplus c_{1,j} \oplus (2 \bullet c_{2,j}) \oplus (3 \bullet c_{3,j})$$
$$S'_{3,j} = (3 \bullet c_{0,j}) \oplus c_{1,j} \oplus c_{2,j} \oplus (2 \bullet c_{3,j})$$

Where, • signifies Finite Field of multiplication over the finite field GF $(2^8)$.

In the AES decryption process, with the help of different set of data values the inverse MixColumn steps are performed.



$$\begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} X \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} = \begin{bmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,0} & b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,0} & b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,0} & b_{3,1} & b_{3,2} & b_{3,3} \end{bmatrix}$$

**Fig 4. MixColumn stages of AES**

### D. Add Round Key Transformation

The fourth step of AES algorithm is AddRoundKey. Here, 128 data bits are XORed with 128 bit of round key bitwise. The column wise operation where 4 bytes of data of a single column and one word of round key is under the operation. [2, 4]. From the initial key using the AES key scheduling algorithm produces a number of round keys.

The initial round of AES uses the initial key as the input for the AddRoundKey operation.

In the AES decryption step, the XOR operation is used. [4,5]

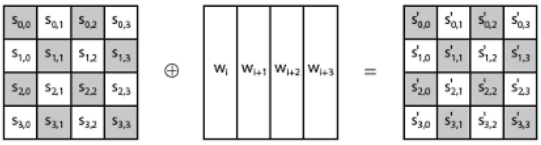

$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,0}$ $s_{1,1}$ $s_{1,2}$ $s_{1,3}$
$s_{2,0}$ $s_{2,1}$ $s_{2,2}$ $s_{2,3}$
$s_{3,0}$ $s_{3,1}$ $s_{3,2}$ $s_{3,3}$
⊕
$w_i$ $w_{i+1}$ $w_{i+2}$ $w_{i+3}$
=
$s'_{0,0}$ $s'_{0,1}$ $s'_{0,2}$ $s'_{0,3}$
$s'_{1,0}$ $s'_{1,1}$ $s'_{1,2}$ $s'_{1,3}$
$s'_{2,0}$ $s'_{2,1}$ $s'_{2,2}$ $s'_{2,3}$
$s'_{3,0}$ $s'_{3,1}$ $s'_{3,2}$ $s'_{3,3}$

**Fig 5. AddRoundKey of AES**

### E. AES Key Expansion

The AES key expansion process takes an input of a 4-word key. It produces a linear array of 44 words. Each round uses 4 of these words. Each word consist of 32 bytes. Every subkey is 128 bits long. The key is copied into the first four words of the expanded key. At a single instance the remainder of the expanded key fill with the four words. Each added word w[i] depends on the immediately preceding word, w [i − 1].[3,5]

$w_i = w_{i-1} \oplus$ wi-4 for all values of i

The result $r_{sk}$ is XORed with $W_{4k-4}$ and a round constant $r_{conk}$

$w_{4k} = r_{sk} \oplus w_{4k-4} \oplus r_{conk}$

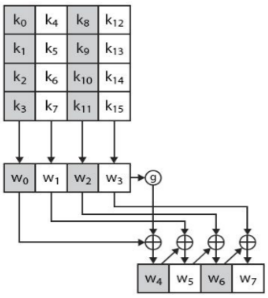

$k_0$ $k_4$ $k_8$ $k_{12}$
$k_1$ $k_5$ $k_9$ $k_{13}$
$k_2$ $k_6$ $k_{10}$ $k_{14}$
$k_3$ $k_7$ $k_{11}$ $k_{15}$
$w_0$ $w_1$ $w_2$ $w_3$
g
$w_4$ $w_5$ $w_6$ $w_7$

**Fig 6. AES Key Expansion**

## III. THE SECURE HASH ALGORITHM SHA-256 OF SHA-2 FAMILY

After SHA-1, the SHA-2 secure hash function is designed by, NSA, USA. It is one of the powerful hash functions available. In comparison with SHA-1, SHA-256 is not much complex.[3] SHA-256 digest value belongs to the SHA-2 family. The 256-bit key makes it a good compatible-function for AES. SHA-256 generates an almost-unique 256-bit (32-byte) signature for a text. [4] here, are the two main components of SHA are to be describe:

a. The function for compression in SHA-256, and

b. The message scheduling function of SHA-256. [3]

The initial hash value H(0) is the following sequence of 32-bit words (which are obtained by taking the fractional parts of the square roots of the first eight primes):

$H_1^{(0)}$ , $H_2^{(0)}$, $H_3^{(0)}$, $H_4^{(0)}$, $H_5^{(0)}$, $H_6^{(0)}$, $H_7^{(0)}$, $H_8^{(0)}$

The hash computation proceeds as follows:

For all the padded message blocks

First initialize the registers a; b; c; d; e; f ; g; h with the intermediate hash value ( the initial hash value when i = 1)

$a = H_1^{(i-1)}$ , $b= H_1^{(i-2)}$  and so on.

Apply the SHA-256 compression functions

Compute Ch(e,f,g), Maj(a,b,c), $\Sigma_0(a)$, $\Sigma_1(e)$ and $W_j$

$T_1 \leftarrow h + \Sigma_1(e) + Ch(e,f,g) + K_j + W_j$

$T_2 \leftarrow \Sigma_0(a) + Maj(a,b,c)$

H ←g

g← f

f← e

e← d+ $T_1$

d← c

c← b

b←a

a← $T_1 + T_2$.

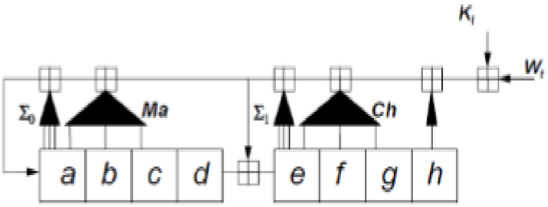


**Fig 7. Hash computation, state register update function**

SHA 256 message schedule procedure is as follows

Wj= Mj(i) for the first 16 blocks  then for rest of the blocks use the following recurrence formula

$W_j = \sigma_1 (W_{j-2}) + (W_{j-7}) + \sigma_0 (W_{j-15}) + (W_{j-16})$

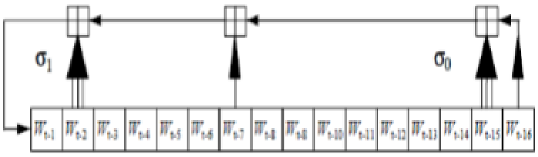

$\sigma_1$
$\sigma_0$
$W_{t-1}$ $W_{t-2}$ $W_{t-3}$ $W_{t-4}$ $W_{t-5}$ $W_{t-6}$ $W_{t-7}$ $W_{t-8}$ $W_{t-9}$ $W_{t-10}$ $W_{t-11}$ $W_{t-12}$ $W_{t-13}$ $W_{t-14}$ $W_{t-15}$ $W_{t-16}$

**Fig 8. (Wj) message scheduled recurrene**

When hashing procedure is completed for all the consecutive 512-bit message blocks, then the last intermediate hash value is the declared as the final overall hash value [3].

SHA-256 is mainly accept the messages with arbitrary lengths.

**The followings are the steps for SHA-256**

Step 1: Append message padding bits
Step 2: Add length to the text
Step 3: Initialize a temporary hash buffer
Step 4: Arrange the message in 512-bit blocks, which forms the main data for the algorithm
Step 5: The final state value is taken as the output as the resulting hash value.

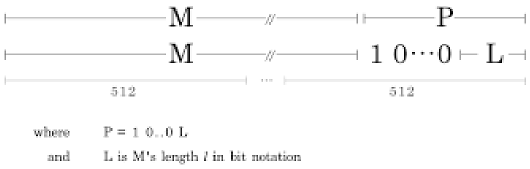


**Fig 9. SHA-256 structure**

## IV. PIXEL VALUE DIFFERENCING METHOD (PVD METHOD)

In the pixel value differencing method [6], the cover image is generally a grey level image and different size secret message bits are used as secret data. Through raster scan order the cover image is divided into two non-overlapping neighbouring blocks with size 1X2. Consider $P_i$ and $P_{i+1}$ are the two consecutive pixels on the ith block. The difference value, $d_i$, is measured by $d_i=|P_i–P_{i+1}|$. We take the absolute value of the difference $d_i$ which represents the variation of each data block.

A lower value of $d_i$ signifies the data presence of smooth area, and greater value is in the edge area. To maintain the intensity values of grey scale image the values of $d_i$ is in the range of [0, 255]. The boundary of range R is expressed by [$lower_i$, $upper_i$]. To calculate the sequences of the number of bits (t)to be hidden for the two consecutive pixels depends on the range table. It is calculated as $t = (\log_2 (upper_i – lower_i) + 1 )$. Now, the new generated bit sequence is converted into equivalent decimal value, $t_d$. The new generated difference value ($d_i$') is obtained by using $d_i'=t_d + lower_i$.

The final pixel values are calculated using the following condition,

$$(P_i', P_{i+1}') = \begin{cases} \left(P_i + \left\lceil \frac{m}{2} \right\rceil, P_{i+1} - \left\lfloor \frac{m}{2} \right\rfloor\right), \text{ if } P_i \geq P_{i+1} \text{ and } d_i' > d_i \\ \left(P_i - \left\lfloor \frac{m}{2} \right\rfloor, P_{i+1} + \left\lceil \frac{m}{2} \right\rceil\right), \text{ if } P_i < P_{i+1} \text{ and } d_i' > d_i \\ \left(P_i - \left\lceil \frac{m}{2} \right\rceil, P_{i+1} + \left\lfloor \frac{m}{2} \right\rfloor\right), \text{ if } P_i \geq P_{i+1} \text{ and } d_i' \leq d_i \\ \left(P_i + \left\lceil \frac{m}{2} \right\rceil, P_{i+1} - \left\lfloor \frac{m}{2} \right\rfloor\right), \text{ if } P_i < P_{I+1} \text{ and } d_i' \leq d_i \end{cases},$$

Where, $m=|d_i'-d_i|$.

## V. PROPOSED ALGORITHM FLOWCHART

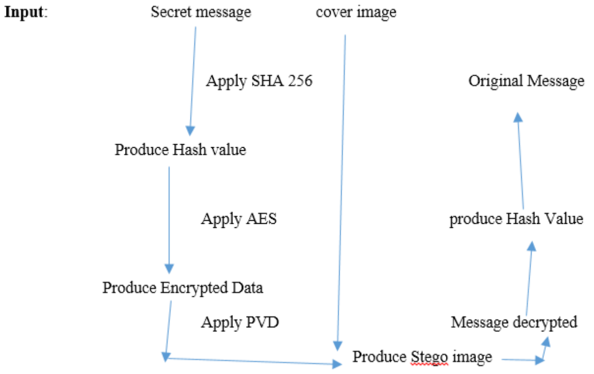


**Fig 10. Proposed algorithm flowchart**

**Proposed Algorithm**

Step 1. Input: Secret Message, Cover Image

Step 2. Use SHA 2 for message preparation

Step 3.

a. Apply AES 128 with key size 256 for data encryption

b. After encryption encrypted data is produced.

Step 4. Now using PVD message hide into cover image and produce Stego image

## VI. RESULT ANALYSIS

Input: plaintext using AES 256 = "Hello India"

Generated Cipher Text = 1FA9 3EA9 E182 9AB6 13EA E92B C271 169F 2AE5 9A2B EA45 31BA A4D2 1B9A 6AC1 9AD0

SHA-256 Input =" Good Morning"

SHA-2 Hashing code = 99e3 3643 42c3 2c90 6d8d a217 4199 9e55 17a1 9ed2 70e4 4524 8947 4ec1 d4b0 1bf3

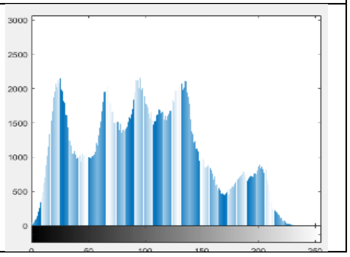
I/P KEY=HASH CODE = 99e3 3643 42c3 2c90 6d8d a217 4199 9e55 17a1 9ed2 70e4 4524 8947 4ec1 d4b0 1bf3

Table 2. The PSNR and MSE analysis of cover image and stego image

### a. After data embedding

| Image name | Size | Cover Image | Stego Image | PSNR(dB) | MSE |
|---|---|---|---|---|---|
| Lena | 512X512 | | | 43.4273 | 3.184 |
| Baboon | 512X512 | | | 37.5427 | 6.239 |
| Boat | 512X512 | | | 42.3278 | 6.984 |

### b. Histogram Analysis

| Image name | Analysis of Histogram |
|---|---|
| Lena |  |

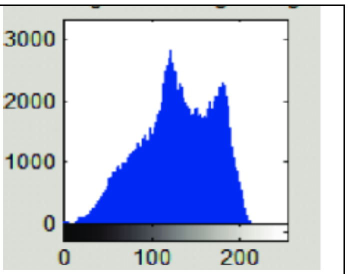
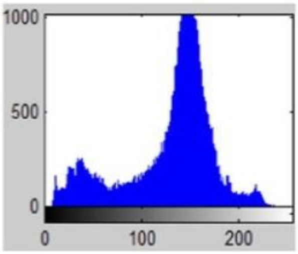
| | |
|---|---|
| Baboon |  |
| Boat |  |

**Fig 11. The analysis of Histogram of sample images**

## VII. CONCLUSIONS

To combine the hashing algorithm SHA 256 and cryptography technique AES 256 in a same model to achieve a better data security to develop a hybrid cryptosystem in terms of complexity. The steganography method along with this hybrid cryptosystem also increase security aspects.

## REFERENCES

1. William Stallings, “Cryptography and Network Security Principles and Practices, Fifth Edition”, Prentice Hall, 2011.
2. FIPS 197, “Advanced Encryption Standard (AES)”, November 26, 2001.
3. M. K. R. Danda, DESIGN AND ANALYSIS OF HASH FUNCTIONS, 2007.
4. R.Nivedhitha, Dr.T.Meyyappan, “Image Security using Steganography and Cryptographic Techniques”, International Journal of Engineering Trends and Technology, Vol.7, pp. 366- 371, 2012.
5. Dr. Helena Handschuh, Dr. Henri Gilbert, “Security Level of Cryptography -SHA-256”, Issy-les-Moulineaux 31 January 2002.
6. Wu D-C, Tsai W-H.2003 A steganographic method for images by pixel value differencing. Pattern Recognit. Lett.24, 1613–1626.(doi:10.1016/S01678655(02)00402-6)
7. Mandal JK,Das D.2012 Steganography using adaptive pixel value differencing(APVD)ofgray images through exclusion of overflow/underflow. In 2nd Int .Conf. on ComputerScience, Engineering and Applications (CCSEA-2012),Delhi,India
8. Hosam O,Halima NB.2016 Adaptive block-based pixel value differencing steganography. Secur. Commun.Netw.9, 5036–5505.(doi:10.1002/sec. 1676)

## AUTHORS PROFILE

**Jayeeta Majumder** was born in Kolkata, India. She received the M.Tech degree in Computer Science & Engineering from West Bengal University of Technology, India, 2009 and currently pursuing Ph D. in Computer Science & Engineering from KIIT University, Odissa, India. In 2009 she joined the Department of Computer Science & Engineering, Haldia Institute of Technology as a lecturer, and in 2013 she became an Assistant Professor. Her currently research area includes Image Processing, Steganography, Cryptography. She has more than 15 publication in reputed National and International Journals. She has some experience as an Assembly programmer and Android Developer. She has the membership of CSI.

**Dr. Chittaranjan Pradhan** was born in Odissa, India. He received the M,Tech degree in Computer Science & Engineering from KIIT University, Bhubaneswar, India and he completed the Ph D. in Computer Science & Engineering From KIIT University, India. In 2013 he joined the Department of Computer Science & Engineering, KIIT University as an Assistant Professor and in 2019 he became an Associate Professor. His research interest includes Image Processing, Digital watermarking, Data analytics, Machine Learning. He organized many International Conferences as an Organizing Committee member. He also reviewed many International Journal. He has total 13 years Academic teaching experience with more than 60 publications in peer reviewed National and International Journals, books & Conferences like Taylor & Francis Springer, Elsevier Science Direct, Inderscience, Annals of Computer Science, Poland, and IEEE. As an author he published many book and edited books of different publishing houses like, Springer, IGI GLOBAL. He is also member of different Professional societies globally in the engineering field.