**International Journal of Engineering Research and Development**
*e-ISSN: 2278-067X, p-ISSN: 2278-800X, www.ijerd.com*
*Volume 11, Issue 09 (September 2015), PP.18-26*

# Secure Image Transmission for Cloud Storage System Using Hybrid Scheme

Shelly[1], Dr Rajesh Kumar Bawa[2]
[1]*Student of Punjabi University, Patiala.*
[2]*Proff. Of Punjabi University, Patiala.*

**Abstract** :- Data over the cloud is transferred or transmitted between servers and users. Privacy of that data is very important as it belongs to personal information. If data get hacked by the hacker, can be used to defame a person's social data. Sometimes delay are held during data transmission. i.e. Mobile communication, bandwidth is low. Hence compression algorithms are proposed for fast and efficient transmission, encryption is used for security purposes and blurring is used by providing additional layers of security. These algorithms are hybridized for having a robust and efficient security and transmission over cloud storage system.

**Keywords:-** Secure Image Transfer, Encryption, Blur Map Model, Compression.

## I. INTRODUCTION

Every user require secure storage and safe data transmission. Several trends are there in cloud computing, which are an internet based deployment. The more powerful processors together with the (SaaS) software as a service computing architecture, transforming data centers into pools of service on large scale. To subscribe high quality services we have to increase network bandwidth, yet flexible connections can be made. As we know that data security is an important term of (QOS) quality of service, cloud computing has to invent new challenging security threats for number of reasons. firstly, traditional cryptographic primitives are the most basic blocks which are used to build cryptographic protocols and security. various kind of user's data is stored on cloud and demand for secure and safe data for long term and verifying the correctness of that data in the cloud becomes even more challenging. Secondly, cloud computing is not just a third party data warehouse. The data stored in the cloud is continuously changing by including insertions, deletion , modification , appending, recording . basically the stored data is frequently updated by users. It is the paramount important to ensure the correctness of data under such dynamic data updates. However, this dynamic feature also makes the traditional integrity insurance technique futile and entails new solutions. User's data is redundantly stored on multiple physical locations to further reduce the data integrity threads. Last, but not the least, deployment of cloud computing is powered by data centers running in the cooperated, simultaneous and distributed manner. Here, distributed are used to ensure the correctness of data in cloud for achieving the secure and robust cloud data storage system in real world.

## II. SECURITY ON IMAGES

Our first goal in this project is the image compression. Various compression schemes have been studied under the first objective. The major compression schemes evaluated under the preliminary study for this research are DFT (Discrete Fourier Transformation), DCT (Discrete Cosine Transformation) and DWT (Discrete Wavelet Transformation) because of their popularity and effectiveness.For images, the JPEG images are taken into account as it preferred DWT over DCT or DFT. In DFT,[6][7] execution time is lower and it provides lower compression as compare to the other techniques. In DCT is simple compression algorithm, because computation count in this algorithm is limited, hence provides lower compression ratio. DWT on the other hand, is complex and computation count is very high and it provides higher compression ratio as compared to later two and also proven to be more effective. In wavelet transform system the entire image is transformed and compressed as a single data object rather than block by block as in a DCT based compression system. It can provide better image quality than DCT, especially on higher compression ratio. After preliminary study of literature based on these compression techniques we evaluated that DWT with HAAR Wavelet is the best performer among all other compression techniques available in our selection in terms of compression ratio and elapsed time. Finally, the decision is made to use DWT for its effectiveness and robustness over DCT and DFT.[6][7].

**2.1 Image Compression Using DWT**

When an image has been processed of the DWT, the total number of transform coefficients is equal to the number of samples in the original image, but the important visual information is concentrated in a few coefficients. To reduce the number of bits needed to represent the transform, all the sub bands are quantized. Quantization of DWT sub bands is one of the main sources of information loss. In the JPEG2000 standard, the quantization is performed by uniform scalar quantization with dead-zone about the origin. In dead-zone scalar quantizer with step-size $\Delta_j$, the width of the dead-zone is $2\Delta_j$ as shown in Figure below. The standard supports separate quantization step-sizes for each sub band. The quantization step size $\Delta_j$ for a sub band $j$ is calculated based on the dynamic range of the sub band values. The formula of uniform scalar quantization with a dead-zone is

$$q_j(m,n) = \text{sign}(y_j(m,n)) \left\lfloor \frac{|W_j(m,n)|}{\Delta_j} \right\rfloor$$

where $W_j(m,n)$ is a DWT coefficient in sub band $j$ and $\Delta_j$ is the quantization step size for the subband $j$. All the resulting quantized DWT coefficients $q_j(m,n)$ are signed integers.

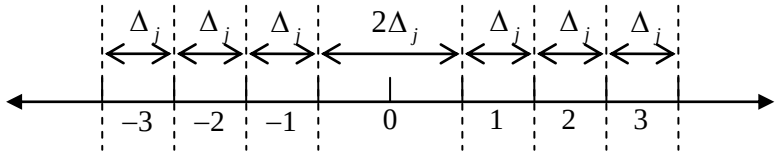


**Figure 1 Dead-zone quantization about the origin.**

After the quantization, the quantized DWT coefficients are then use entropy coding to remove the coding redundancy.

**2.2 Image Encryption using Blowfish**

To perform the encryption in the second object, blowfish encryption algorithm is used to hide the image details of hidden object.[1,3-4,8] A significant number of research papers on the performance evaluation and work flow of encryption algorithms has been studies under the literature survey part. The AES and Blowfish algorithms were selected in the final short listing of encryption algorithms, because these two provide the best encryption security. Out of the two shortlisted ones, the conclusion was obtained that the blowfish encryption algorithm is considered the fastest one among the all other options. [14] Blowfish encryption algorithm is designed in a customized way to work with images in MATLAB environment. The algorithm code is designed to perform various rounds of encryption. The encryption algorithm is used here to hide the image details and to create a new image with dizzy image details. The image details are made hidden in chaotic way to create a new image with less number of details. The image is not made completely unreadable because it provokes the hacker to crack into the encryption, whereas a low resolution less detail encryption can be easily mistaken as a bad image.. The decryption process is the reverse process, which is used to obtain the original image by using the reverse engineering of the cryptographic process on the receiver's end. For the decryption, user has to enter the same key as it was entered on the sender's side while encrypting the image. The decryption process returns the full resolution original image from the encrypted image once the process is complete. The image encryption using blowfish process has been listed in the figure below(see Figure 2.

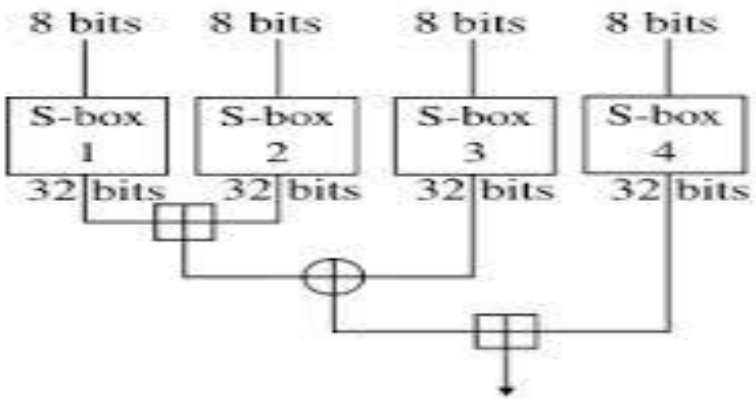


**Figure 2 Blowfish encryption for image processing**

Blowfish is a variable-length key, 64-bit block cipher. The algorithm consists of two parts: a key-expansion part and a data- encryption part. Key expansion converts a variable-length key of at most 56 bytes (448 bits) into several sub key arrays totaling 4168 bytes. Data encryption occurs via a 16-round Feistel network. Each round consists of a key-dependent permutation, and a key- and data-dependent substitution. The additional operations are four indexed array data lookups per round. Implementations of Blowfish that require the fastest speeds should unroll the loop and ensure that all sub keys are stored in cache.

### 1.1.1 Sub keys:

Blowfish uses a large number of sub keys. These keys must be precomputed before any data encryption or decryption.

1. The P-array consists of 18 32-bit sub keys: P1, P2,..., P18.
2. There are four 32-bit S-boxes with 256 entries each:

S1,0, S1,1,..., S1,255;
S2,0, S2,1,..,, S2,255;
S3,0, S3,1,..,, S3,255;
S4,0, S4,1,..,, S4,255.

**Generating the Sub keys:**

1. The sub keys are calculated using the Blowfish algorithm. The exact method is as follows:
2. Initialize first the P-array and then the four S-boxes, in order, with a fixed string. This string consists of the hexadecimal digits of pi (less the initial 3). For example:
P1 = 0x243f6a88, P2 = 0x85a308d3, P3 = 0x13198a2e, P4 = 0x03707344
2. XOR P1 with the first 32 bits of the key, XOR P2 with the second 32-bits of the key, and so on for all bits of the key (possibly up to P14). Repeatedly cycle through the key bits until the entire P-array has been XORed with key bits.
3. Encrypt the all-zero string with the Blowfish algorithm, using the sub keys described in steps (1) and (2).

4. Replace P1 and P2 with the output of step (3).
5. Encrypt the output of step (3) using the Blowfish algorithm with the modified sub keys. 6. Replace P3 and P4 with the output of step (5).
6. Continue the process, replacing all entries of the P- array, and then all four S-boxes, with the output of the continuo

There are a number of building blocks that have been demonstrated to produce strong ciphers.

- Large S-boxes: Larger S-boxes are more resistant to differential cryptanalysis. An algorithm with a 32-bit word length can use 32-bit S-boxes.
- Key-dependent S-boxes: key-dependent S-boxes are much more resistant to these attacks differential and linear cryptanalysis.
- Combining operations: Combining XOR mod $2^{16}$, addition mod $2^{16}$, and multiplication mod $2^{16}+1$ [7].

Key-dependent permutations: The fixed initial and final permutations of DES have been long regarded as cryptographically worthless.

### 2.3 Blurring Algorithm

The proposed scheme is a modification of the one suggested by Lian et al. In their blurring algorithm, an explicit diffusion function based on a logistic map is used to spread out the influence of a single plain image pixel over many cipher image elements. Although the diffusion function is executed at a fairly high rate, it is still the highest cost, in terms computational time, of the whole blurring algorithm. This is because multiplications of real numbers are required in this stage. Table 1 lists the time required in different parts of Lian et al's blurring algorithm. It shows that the time for completing a single diffusion round is more than four times longer than that for a permutation. The test is performed on a personal computer (PC) with 2.0GHz Core i3 processor, 6 GB memory and 500GB harddisk capacity.

In our modified confusion stage, the new position of a pixel is calculated according to Eq. (1). Before performing the pixel relocation, diffusion effect is injected by adding the current pixel value of the plain image to the previous permuted pixel and then performs a cyclic shift. Other simple logical operations such as XOR can be used instead of the addition operation. The shift operation can also be performed before addition. However, simulation results found that the "add and then shift" combination leads to the best performance and so it becomes the choice in our blurring algorithm. The new pixel value is then given by Eq. (5). [ ] ( )mod , ( ) i = i + i−1 3 i−1 v Cyc p v L LSB v (5) where pi is the current pixel value of the plain image, L is the total number of gray levels of the image, vi-1 is the value of the (i-1)th pixel after permutation, Cyc[s, q] is the q-bit

right cyclic shift on the binary sequence s, LSB3(s) refers to the value of the least three significant bits of s, vi is the resultant pixel value in the permuted image. The seed [ ] 0, 1 v−1 ∈ L − is a sub-key obtained from the key generator.

As the pixel value mixing depends on the value of the previously processed pixel, the operation sequence cannot be changed. This may lead to a problem in the reversed confusion process required in deblurring. A solution is to make the first decipher round perform the reverse position permutation only. Then both reverse position permutation and pixel value change are performed from the second decipher round. In this manner, an additional deblur round is required for the reverse of pixel value modification. It composes of the simple add-and-shift operation and adds only little cost to the overall deblur procedures.

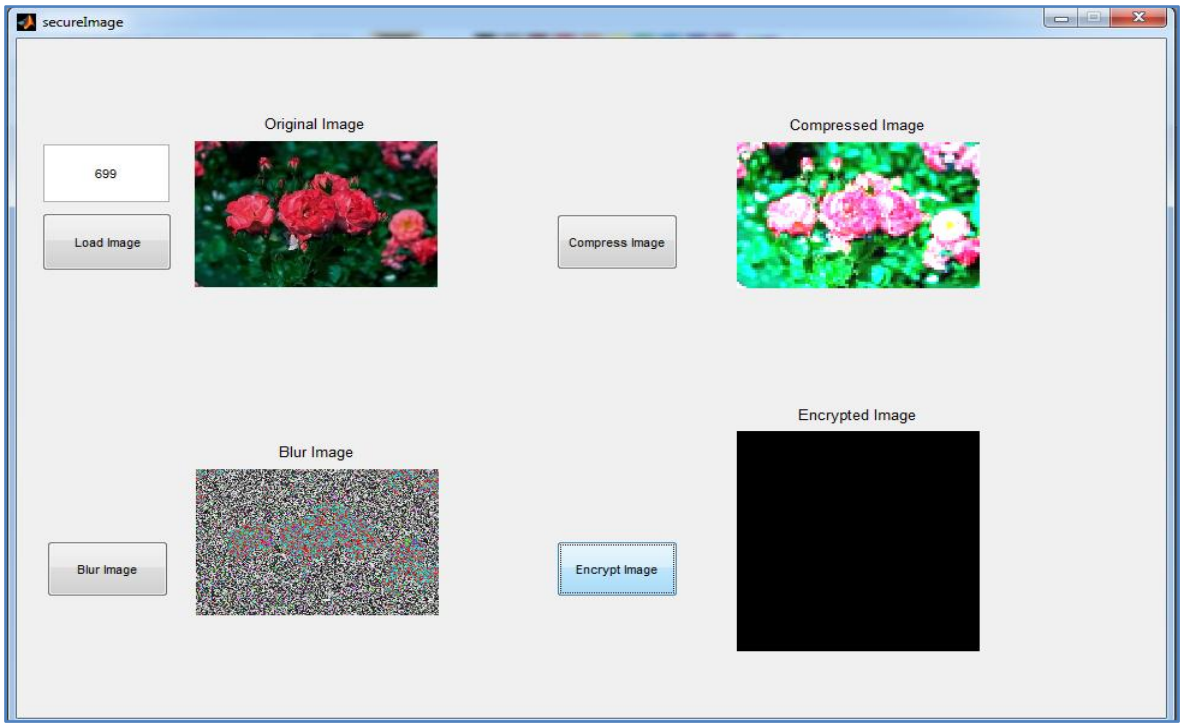


**Figure 3: The step-wise security embedding model using compression, blur and encryption for the proposed system security model to secure the new image and add it to the database**

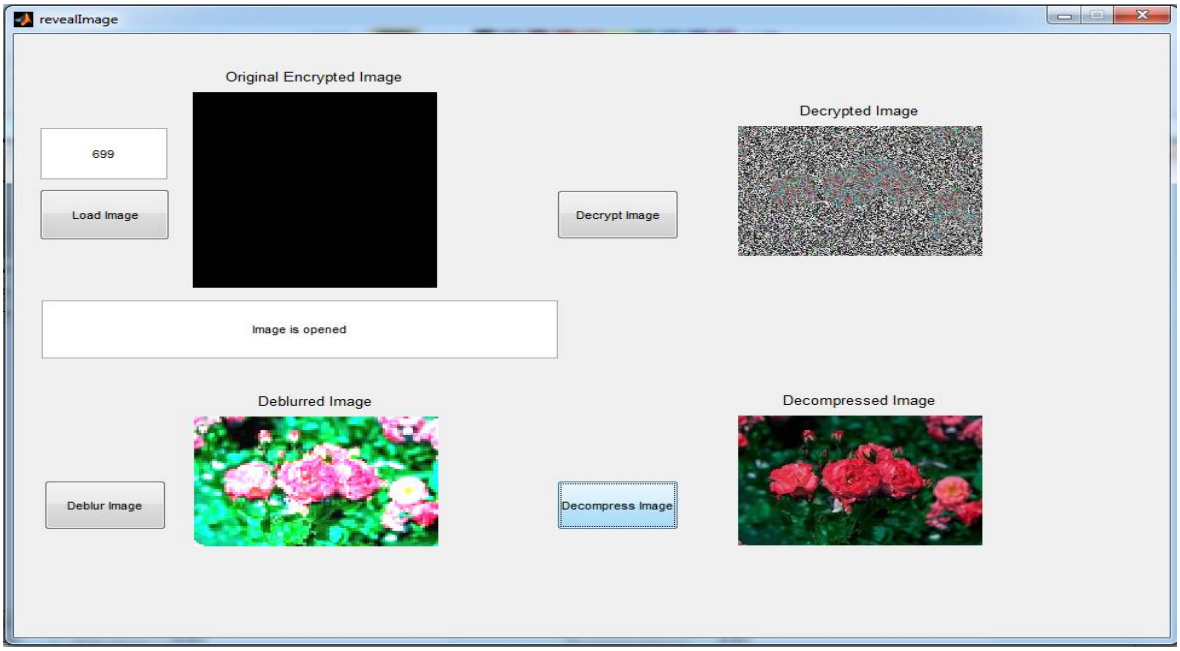


**Figure 4: The step-wise reversal security embedding model using compression, blur and encryption for the proposed system security model to reveal the secured image.**

| Index | File Size (Kb) | PROPOSED SCHEME | | EXISTING SCHEME | |
|---|---|---|---|---|---|
| | | Encryption Time (seconds) | Decryption Time (seconds) | Encryption Time (seconds) | Decryption Time (Seconds) |
| **1** | 1183.711 | 2.565694 | 0.646096 | 11.60778 | 10.28958 |
| **2** | 811.6875 | 1.736718 | 0.625936 | 7.973028 | 7.549324 |
| **3** | 689.6484 | 1.454134 | 0.62148 | 7.106211 | 6.66421 |
| **4** | 585.7813 | 1.236788 | 0.626178 | 5.839042 | 5.249151 |
| **5** | 930.1094 | 1.958269 | 0.623342 | 9.336228 | 8.407667 |
| **6** | 1481.406 | 3.086175 | 0.622886 | 14.74685 | 13.03476 |
| **7** | 464.9531 | 0.988719 | 0.626737 | 4.742196 | 4.296639 |
| **8** | 1106.797 | 2.322694 | 0.62038 | 11.07452 | 10.25855 |
| **9** | 339.4688 | 0.724391 | 0.623164 | 3.458914 | 3.13864 |
| **10** | 793.4063 | 1.656799 | 0.626932 | 8.34238 | 7.231282 |
| **11** | 1347.328 | 2.822294 | 0.622072 | 13.83517 | 12.14879 |
| **12** | 988.3672 | 2.083992 | 0.624786 | 10.15554 | 8.96825 |
| **13** | 987.7813 | 2.076051 | 0.6259 | 10.25987 | 9.525794 |
| **14** | 837.7031 | 1.760459 | 0.624406 | 8.473982 | 7.558881 |
| **15** | 632.4609 | 1.333392 | 0.619688 | 6.39154 | 5.502444 |
| **16** | 1370.156 | 2.880993 | 0.623049 | 13.88179 | 12.39352 |
| **17** | 861.3281 | 1.818985 | 0.625489 | 8.544957 | 7.559276 |
| **18** | 1215.5 | 2.54242 | 0.623796 | 12.44276 | 10.8589 |
| **19** | 889.875 | 1.896367 | 0.62476 | 9.271283 | 8.295494 |
| **20** | 827.3828 | 1.740709 | 0.623639 | 8.228253 | 7.261873 |
| **21** | 1276.133 | 2.663857 | 0.622935 | 13.1867 | 11.80048 |
| **22** | 1403.5 | 2.94659 | 0.625573 | 14.31333 | 12.92009 |
| **23** | 1127.906 | 1.074278 | 0.622783 | 5.285584 | 5.416851 |
| **24** | 883.5469 | 1.860694 | 0.623106 | 10.16153 | 7.988927 |
| **25** | 1825.641 | 3.834191 | 0.625366 | 18.813 | 17.10785 |
| **26** | 1145.063 | 2.398937 | 0.623667 | 11.61872 | 10.07544 |
| **27** | 792.9297 | 1.675993 | 0.625483 | 8.936339 | 6.941237 |
| **28** | 797.0156 | 1.677833 | 0.626425 | 7.910779 | 7.030197 |
| **29** | 986 | 2.06346 | 0.625672 | 9.986034 | 9.198403 |
| **30** | 548.4375 | 1.157437 | 0.622987 | 5.257926 | 4.652692 |

**Table 1: The table displaying the results of improved BLOWFISH Implementation on dataset of 50 images**

All measurements were taken on a single core of an Intel Core i3-2400 CPU at 3100 MHz, and averaged over 10000 repetitions. Our findings are summarized in Table 1 One can see that while the initialization overhead generally has a huge impact on the performance, this effect starts to fade out already at messages of around 256-1500 bytes. Due to the fast implementation of Blowfish algorithm, it has performed way better than the existing AES encryption methods available. The proposed algorithm achieves nearly optimal performance starting from 512 byte message length due to its ability of programming structure which enables it to fully utilize the improved multiple encryption patterns and validation for its initialization overhead. The proposed algorithm has generally performs better than the existing when configured with block size of 128-bit and fixed s-box implementation. Also the validation method has been added to provide more flexibility and robustness to the proposed algorithm.

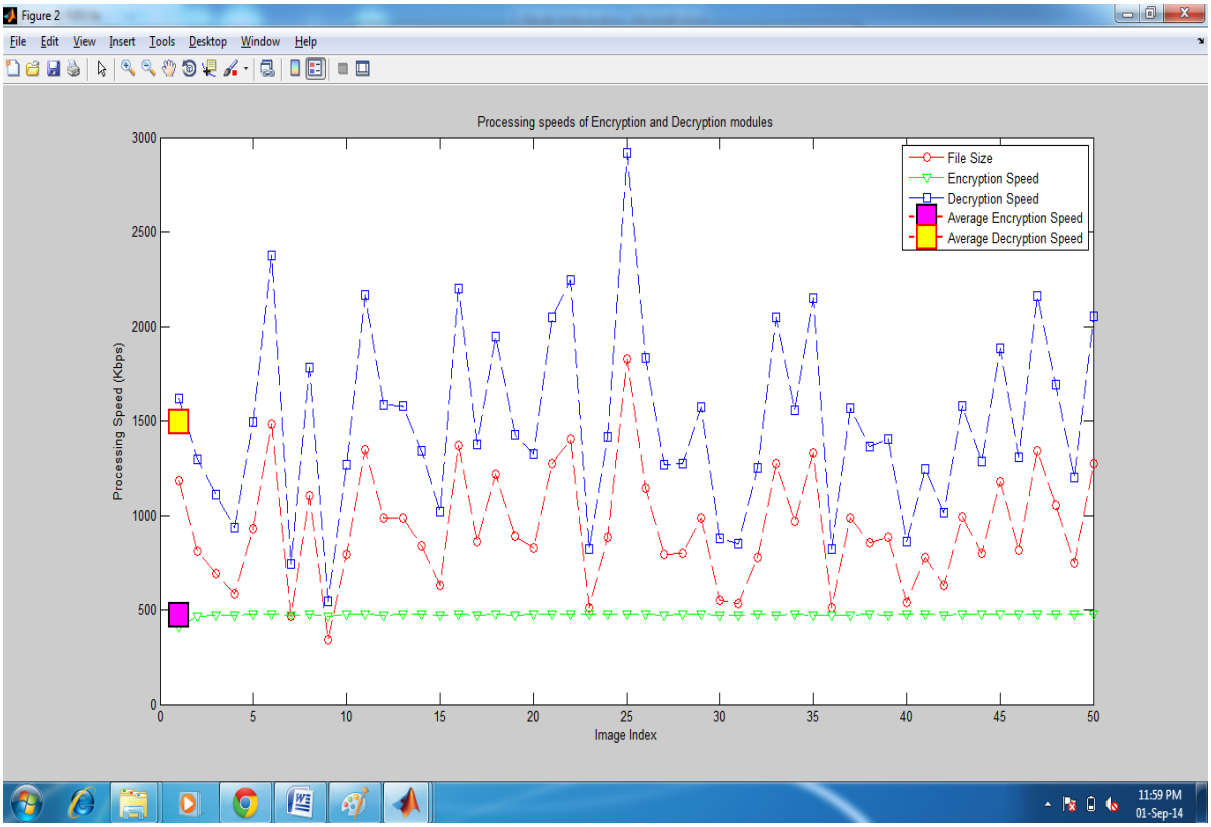


**Figure 5: The graphs of Encryption and Decryption processing speeds [Also the average results] for proposed model**

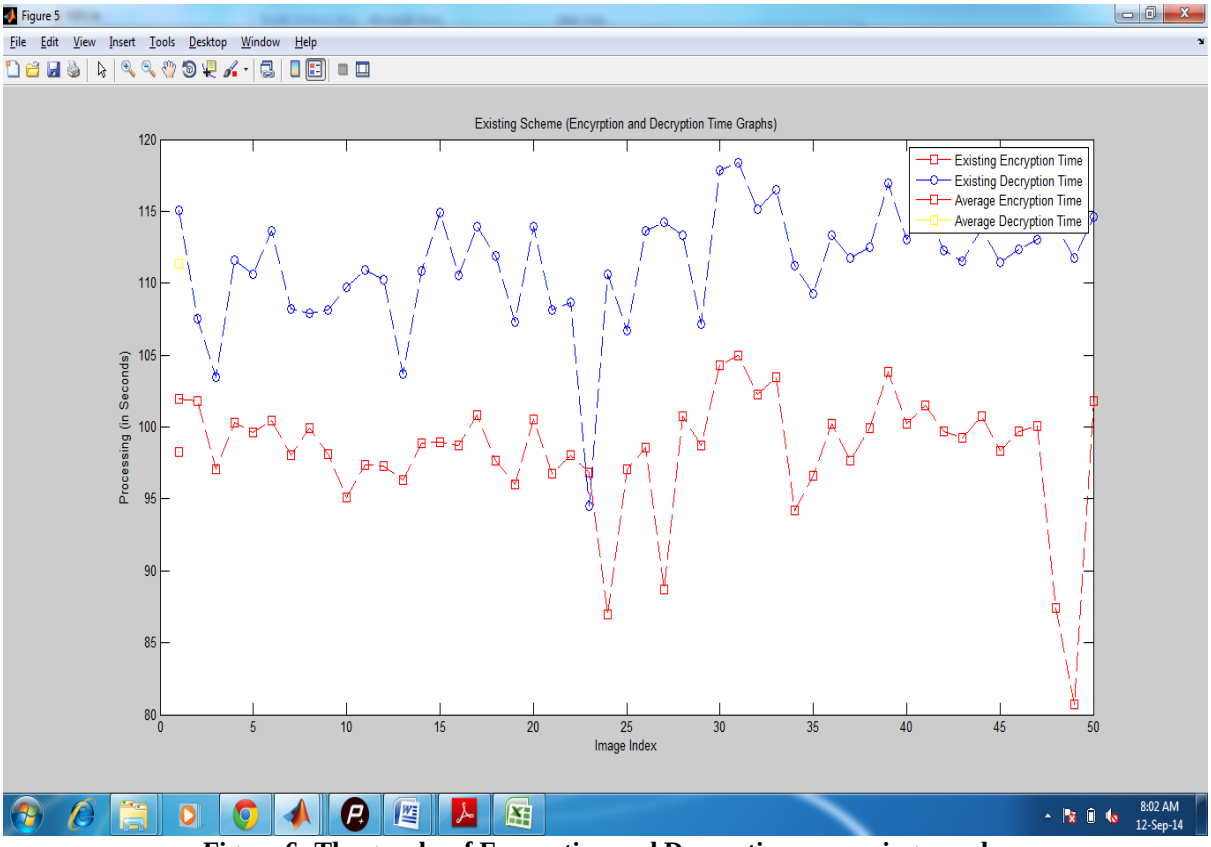


**Figure 6: The graphs of Encryption and Decryption processing speeds [Also the average results] for existing model**

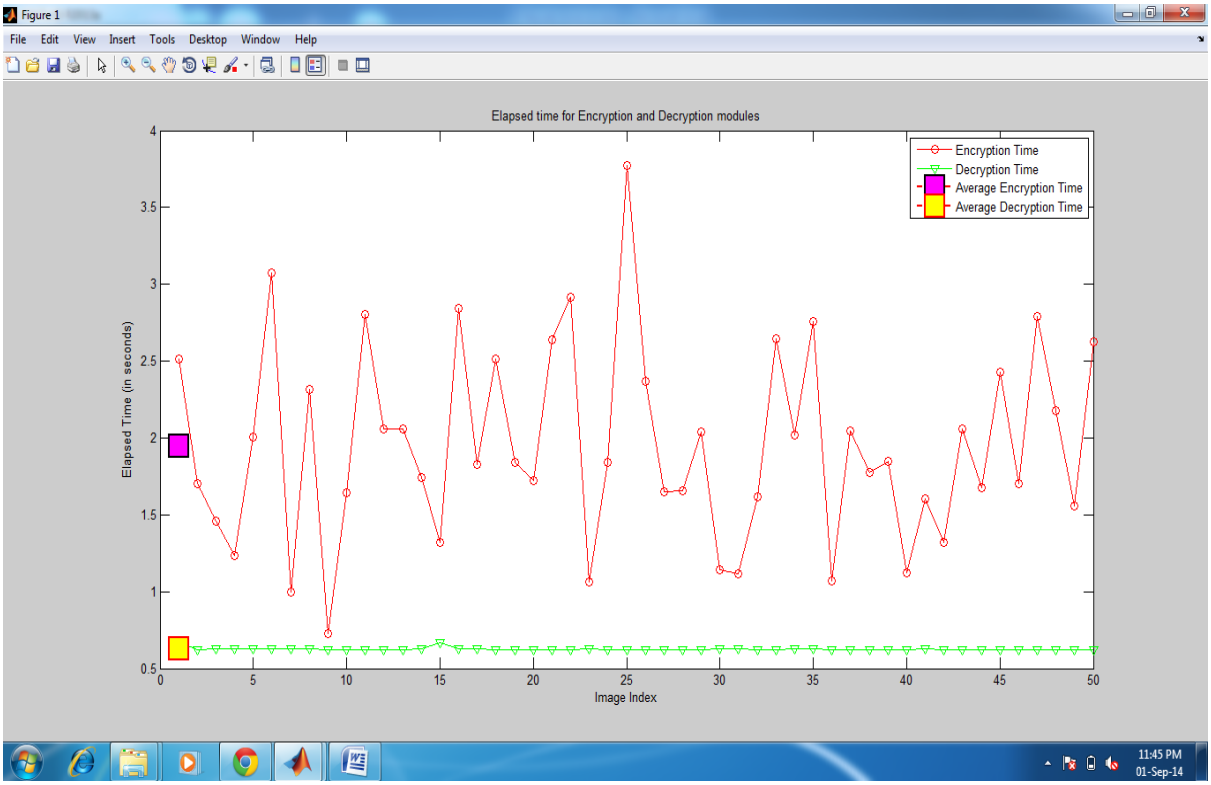


**Figure 7: The graphs of Encryption and Decryption time of proposed system**

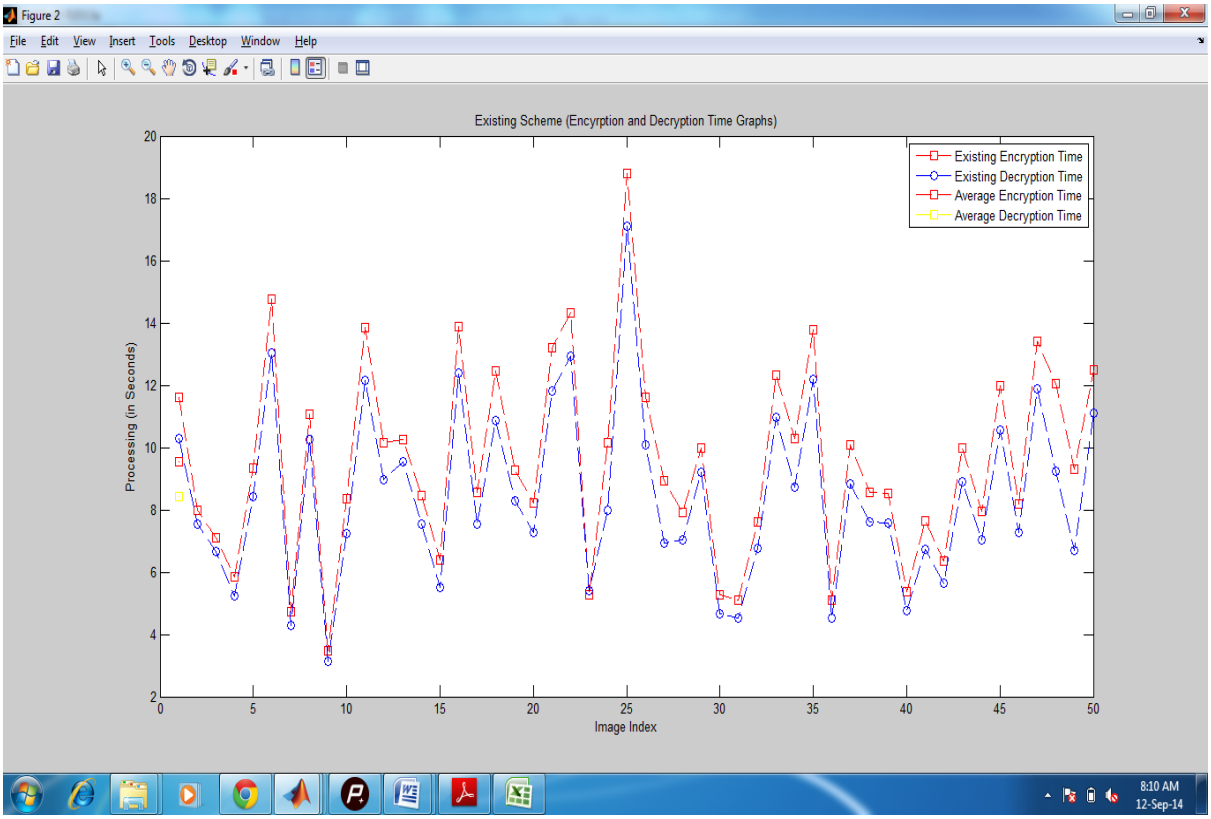


**Figure 8: The graphs of Encryption and Decryption time of existing system**

The proposed algorithm has been proved to be way faster than the existing AES algorithm. The existing algorithm is taking almost 2-3 times slower than the proposed algorithm. The proposed algorithm has been proved to be efficient for both image and text data.

## III. CONCLUSIONS

In this paper, we are proposing the compression, encryption and blurring security techniques by using DWT , BLOWFISH and BLURMAP [ providing extra layers of security] and proposed hybridized algorithm technique for more secure and effective security system. Blowfish is compared with AES technique which proves that Blowfish is best encryption technique in all way.

## ACKNOWLEDGEMENT

I deeply thankful to Dr Rajesh Kumar Bawa , professor , Punjabi University, Patiala. For providing his valuable help throughout my work. I am thankful for his stimulating guidance, continuous encouragement, and supervision. And with all hard work I completed my Research Paper.

## REFERENCES

[1]. Al-Hilo, Eman A., and Rusul Zehwar. "Fractal Image Compression by YIQ Color Space." In *Computational Science and Computational Intelligence (CSCI), 2014 International Conference on*, vol. 1, pp. 221-225. IEEE, 2014.

[2]. Xiangui Kang, Jiwu Huang [2003]: "A DWT-DFT composite watermarking scheme robust to both affine transform and JPEG compression" IEEE aug 2003.

[3]. Sonja Grgic, Mislav Grgic [2001]: **"**Performance analysis of image compression using wavelets." Industrial elsctronics. IEEE 2001.

[4]. Marcelloni, Francesco, and Massimo Vecchio. "A simple algorithm for data compression in wireless sensor networks." *Communications Letters, IEEE* 12, no. 6 (2008): 411-413.

[5]. Srisooksai, Tossaporn, Kamol Kaemarungsi, Poonlap Lamsrichan, and Kiyomichi Araki. "Energy-Efficient Data Compression in Clustered Wireless Sensor Networks using Adaptive Arithmetic Coding with Low Updating Cost."*International Journal of Information and Electronics Engineering* 1, no. 1 (2011): 85-93.

[6]. Xi Xu et al. [2012]: Image Compression Using Radon Transform With DCT : Performance Analysis. International Journal of Scientific Engineering and Technology. Volume No. 2, Issue No. 8, pp : 759-765.

[7]. Dong-U Lee et al. [2009]: Precision-Aware Self-Quantizing Hardware Architectures for the Discrete Wavelet Transform, volume 21, IEEE [2009].

[8]. Dolfus, Kirsten, and Torsten Braun. "An evaluation of compression schemes for wireless networks." In *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2010 International Congress on*, pp. 1183-1188. IEEE, 2010

[9]. . A Survey on Data Compression in Wireless Sensor Network Naoto Kimura and Shahram Latifi *Electrical and Computer Engineering, University of Nevada, Las Vegas{naoto, latifi}@egr.unlv.edu* International Conference on Information Technology: Coding and Computing (ITCC'05) IEEE [2000].

[10]. Medeiros, Henry Ponti, Marcos Costa Maciel, Richard Demo Souza, and Marcelo Eduardo Pellenz. "Lightweight Data Compression in Wireless Sensor Networks Using Huffman Coding." *International Journal of Distributed Sensor Networks* 2014 (2014).

[11]. Liang, Yao. "Efficient temporal compression in wireless sensor networks." In*Local Computer Networks (LCN), 2011 IEEE 36th Conference on*, pp. 466-474. IEEE, 2011.

[12]. Marcelloni, Francesco, and Massimo Vecchio. "A simple algorithm for data compression in wireless sensor networks." *Communications Letters, IEEE* 12, no. 6 (2008): 411-413

[13]. Paar, Christof. "Applied cryptography and data security." *Lecture Notes), Ruhr-Universität Bochum (http://www. crypto. ruhr-uni-bochum. de)* (2000).

[14]. Hager, Creighton TR, Scott F. Midkiff, Jung-Min Park, and Thomas L. Martin. "Performance and energy efficiency of block ciphers in personal digital assistants." In *Pervasive Computing and Communications, 2005. PerCom 2005. Third IEEE International Conference on*, pp. 127-136. IEEE, 2005.

[15]. Kessler, Gary C. "An overview of cryptography." (2003).

[16]. Agarwal, Navita, and Himanshu Sharma. "An Efficient Pixel-shuffling Based Approach to Simultaneously Perform Image Compression, Encryption and Steganography." *International Journal of Computer Science and Mobile Computing (IJCSMC)* 2, no. 5 (2013): 376-385.

[17]. Mathur, Milind, and Ayush Kesarwani. "Comparison between Des, 3des, Rc2, Rc6, Blowfish And Aes." In *Proceedings of National Conference on New Horizons in IT-NCNHIT*. 2013.

[18]. Verma, O. P., Ritu Agarwal, Dhiraj Dafouti, and Shobha Tyagi. "Peformance analysis of data encryption algorithms." In *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*, vol. 5, pp. 399-403. IEEE, 2011.

[19]. Abdouli, Ameera Salem, Joonsang Baek, and Chan Yeob Yeun. "Survey on computationally hard

problems and their applications to cryptography." In*Internet Technology and Secured Transactions (ICITST), 2011 International Conference for*, pp. 46-52. IEEE, 2011.

[20]. Kautsky, Jaroslav, Jan Flusser, Barbara Zitová, and Stanislava Šimberová. "A new wavelet-based measure of image focus." Pattern Recognition Letters 23, no. 14 (2002): 1785-1794.

[21]. Welk, Martin, David Theis, and Joachim Weickert. "Variational deblurring of images with uncertain and spatially variant blurs." Lecture notes in computer science 3663 (2005): 485.

[22]. Levin, Anat. "Blind motion deblurring using image statistics." In Advances in Neural Information Processing Systems, pp. 841-848. 2006

[23]. Danielyan, Aram, Vladimir Katkovnik, and Karen Egiazarian. "BM3D frames and variational image deblurring." Image Processing, IEEE Transactions on 21, no. 4 (2012): 1715-1728.

[24]. Whyte, Oliver, Josef Sivic, Andrew Zisserman, and Jean Ponce. "Non-uniform deblurring for shaken images." International journal of computer vision 98, no. 2 (2012): 168-186.

[25]. Chen, Gu L., Ching Yang Lin, Hon Fai Yau, Ming Kuei Kuo, and Chi Ching Chang. "Wave-front reconstruction without twin-image blurring by two arbitrary step digital holograms." Optics express 15, no. 18 (2007): 11601-11607.

[26]. Couzinie-Devy, Florent, Jian Sun, Karteek Alahari, and Jean Ponce. "Learning to estimate and remove non-uniform image blur." In Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, pp. 1075-1082. IEEE, 2013.

[27]. Whyte, Oliver, Josef Sivic, Andrew Zisserman, and Jean Ponce. "1 Efficient, Blind, Spatially-Variant Deblurring for Shaken Images." Motion Deblurring: Algorithms and Systems (2014): 75.