CHAPTER 7
EXPERIMENTAL RESULTS AND PERFORMANCE
EVALUATION OF PROPOSED DIGITAL
DATA PROTECTION FRAMEWORK

7.1 Introduction

In this research work on “Digital Data Protection Using Steganography and Hardware Identification”, four algorithms were developed for providing effective data protection. The results of these algorithms were analyzed and their performance on application platforms like mobile, desktops/laptops and tablets were tested. Since internet is widely used for data exchange and communication, data protection is of utmost importance. Software protection is useful for preventing software piracy and unauthorized access of applications.

Six algorithms were proposed as part of this thesis. Those are,

1. SteganoDB for secure DB storage of data using steganography techniques
2. RGB based new Steganography method which is difficult to detect
3. Desktop/Laptop application protection using Steganography
4. Desktop/Laptop data protection using Steganography
5. Mobile application protection using Steganography
6. Mobile data protection using Steganography

7.2 Image Quality Metrics

Image quality can be defined as characteristic of an image which measures how the image degradation will be perceived when compared to a typically ideal or a perfect image. When any image modifying algorithm is applied whether it is to enhance it or to apply modifications, Imaging systems may introduce little bit of
distortion in the original image. Hence proper quality assessment of an image is important. Typically image quality is measured in terms of a deviation from the ideal situation and quality measures become technical. Image quality can, however, also be related to the subjective perception of an image, e.g., a human looking at a photograph.

Commonly used image steganography metrics are,

1. Mean Square Error (MSE)
2. Peak Signal to Noise Ratio (PSNR)
3. Structural Similarity Index (SSI)

### 7.2.1 Mean Square Error

Mean squared error or MSE is a measurement of the average of the squares of the errors and for this the difference between the estimator and what the estimated is calculated. This difference may occur because of randomness of pixels or because the estimator doesn't account for information that could produce a more accurate estimate.

The Mean Squared Error calculates how much close a fitted line is to the data points. For each of these data points the distance vertically from the first point to the corresponding y value on the curve fits the error and then square the value. After doing this all the data points are added up, and again divided by the number of points minus two. This squaring or calculation is done so that the negative values does not cancel out the positive values. The smaller the Mean Squared Error is that much closer will be the fit of this data. The MSE has the units squared to whatever that is plotted on the vertical axis.

\[
MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2
\]

(7.1)
7.2.2 Peak Signal to Noise Ratio

Peak signal-to-noise ratio or PSNR is the calculated term for finding the ratio between the maximum possible power of a digital signal and the corresponding power of noise which affects the representation of the image. Many of these digital signals have a very wide dynamic range and PSNR is expressed as logarithmic decibel scale. PSNR can measure the quality of decoding of lossy compression methods like jpeg compression used for images. It will compare the original data and the noise introduced by the compression. While comparing the compression codecs that are typically used in images, PSNR uses an approximation of the human perception regarding the image quality. The higher the PSNR value is will generally mean that the image is of higher quality.

PSNR can be easily defined via the mean squared error (MSE). For noise-free $m\times n$ monochrome image $I$ and its noisy approximation $K$, $MSE$ is defined as:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2$$

The PSNR (in dB) is defined as:

$$PSNR = 10 \cdot \log_{10} \left( \frac{MAX_I^2}{MSE} \right)$$

$$= 20 \cdot \log_{10} \left( \frac{MAX_I}{\sqrt{MSE}} \right)$$

$$= 20 \cdot \log_{10} (MAX_I) - 10 \cdot \log_{10} (MSE)$$

(7.3)

Here, $MAX_I$ represents the maximum possible pixel value of the image.

**Legends:**

$I$ stand for the matrix data of our original image.

$K$ stands for the matrix data of our degraded image.

$m$ represents the numbers of rows of pixels of the images and $i$ represents the index of that row.
\( n \) represents the number of columns of pixels of the image and \( j \) represents the index of that column.

If the pixels are represented using 8 bits per sample this becomes 255. Typically color images have 3 RGB values per pixel. In this case, the definition of PSNR is the same except the MSE is the addition of all squared value differences divided by image size and by three. It is also possible for the color image pixels to be converted to a different color space and then PSNR is calculated for each channel of that color space, e.g., YCbCr or HSL. Typically, the values for the PSNR in lossy image like jpeg and for video compression are between 30 and 50 dB, provided the bit depth is 8 bits. For a 16-bit data typical values for the PSNR are between 60 and 80 db.

Example images along with corresponding PSNR values are given below:

\[
\text{PSNR} = 40 \text{ dB} \quad \text{PSNR} = 30 \text{ dB} \quad \text{PSNR} = 20 \text{ dB}
\]

\[
\text{PSNR} = 10 \text{ dB} \quad \text{PSNR} = 0 \text{ dB}
\]

Fig. 7.1 PSNR ratio and noise

Suppose there is no noise. Then the two images \( I \) and \( K \) are identical, and thus the MSE will become zero. In this case the PSNR is infinite (or undefined, see
Division by zero). The main limitation of PSNR metric is that it relies strictly on numeric comparison and does not actually take into account the human eyes or vision system such as the structural similarity index. (SSIM)

7.2.3 Structural Similarity Index

The structural similarity (SSIM) index is useful to measure the similarity between two images at the same time making sure that the results are more consistent with human perception than the other measuring techniques like mean square error (MSE). Blurred images might be seen as bad quality by the human eye, and this will be similar to the results from the SSIM metric and unlike the MSE method, which considers a blurred image as similar to its focused original. Because of its similarity to human perception, SSIM has become an accepted part of image quality as well as video quality especially when analysis of compressed video data.

SSIM index is a full reference metric where the measurement or prediction of image quality is based on an initial uncompressed image that is used as reference. SSIM method is designed to improve on the other methods such as peak signal-to-noise ratio (PSNR) and mean squared error (MSE) because these methods are proven to be inconsistent with how humans perceive the images visually.

The difference between SSIM and the other techniques mentioned such as MSE or PSNR is that these methods calculates the absolute errors while on the other hand, SSIM method is a perception-based model that considers image degradation as a perceived change in structural information of the image. At the same time it will also incorporate important perceptual phenomena, such as the luminance masking and contrast. Structural information is based on the idea that the pixels in the image have strong inter-dependencies. These carry important information about the structure of the objects in the image as in visual scene.
The SSIM index is calculated on various windows of an image. The measure between two windows \( x \) and \( y \) of common size \( N \times N \) is:

\[
\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)}
\]  

(7.4)

with

- \( \mu_x \) the average of \( x \);
- \( \mu_y \) the average of \( y \);
- \( \sigma_x^2 \) the variance of \( x \);
- \( \sigma_y^2 \) the variance of \( y \);
- \( \sigma_{xy} \) the covariance of \( x \) and \( y \);
- \( c_1 = (k_1 L)^2 \), \( c_2 = (k_2 L)^2 \) two variables which stabilize the division with weak denominator;
- \( L \) the dynamic range of the pixel-values (is \( 2^\#\text{bits per pixel} - 1 \));
- \( k_1 = 0.01 \) and \( k_2 = 0.03 \) by default.

The result of the calculation of SSIM index on an image will be a decimal value between -1 and 1, and value 1 will be possible if there are two identical sets of data.

A more advanced form of SSIM, called Multi-Scale SSIM is also developed and in this case the calculation is conducted over multiple process of multiple stages of sub-sampling which is reminiscent of multi scale processing in the early vision system. The performance of both SSIM and SSIM using multiscale is very high with regards to correlations to human judgments, using the measurements on the most widely used public image quality databases.

SSIM method of image quality calculation has excellent performance and extremely low compute cost and due to this SSIM has become quite widely used in
the cable and the television programs where it has been used as a dominant method of measuring video quality in broadcast as well as movie production houses throughout the entertainment industry. SSIM is the most widely used quality measurement used in several of the commercial video/image processing tools also.

Structural dissimilarity (DSSIM) is a distance metric which has been derived from SSIM (where the triangle inequality is not satisfied).

\[
DSSIM(x, y) = \frac{1 - SSIM(x, y)}{2} \tag{7.5}
\]

SSIM results closer to 1 are considered similar, whereas MSE values of 0 also yields the same result.

<table>
<thead>
<tr>
<th>Original Image (Reference)</th>
<th>SSIM</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blurred Image</td>
<td>0.5257</td>
<td>269.469</td>
</tr>
<tr>
<td>Dilated Image</td>
<td>0.6504</td>
<td>769.773</td>
</tr>
</tbody>
</table>

### 7.3 Data Quality Metrics

1. Data Specification
2. Accuracy
3. Timeliness/Availability
4. Relevance/Trust
7.3.1 Data Specification

A data definition specification could be developed for organizations or specialized fields in order to improve the quality of the products they produce through consistency and transparency. This particular standard helps in eliminating redundancy and also provides, standardizes making it easier and more standard to create, change, scrutinize, analyze and provide information across the enterprise.

During the development of these data definitions, the words used must be traceable to clearly defined data.

7.3.2 Accuracy

Accuracy means how close the result of a query is to the true value so that it can be accepted as the correct value. The difference between observed values and true values indicate the accuracy of the value. There are two types of accuracy. They are positional and attribute. Positional accuracy relates to geographic location. Attribute accuracy denotes the characteristics of an object.

7.3.3 Timeliness/Availability

Data may be correctly entered when it is created. But there is a possibility that it could change over a period of time making it inaccurate or incomplete. Maintaining the data over a period of time is an important part of continuous data quality improvement for master data. Timeliness can be defined as when data is requested and when data is expected also when it is readiness for use.

7.3.4 Relevance/Trust

Relevance of data has two-level requirements: 1) the amount of accessed data used by users and 2) the degree to which the data produced matches the needs of the users in the aspects of indicator definition, elements, classification, etc.
7.4 Security Threats

7.4.1 Attacks

Attack is any attempt to destroy, expose, alter, disable, steal or gain unauthorized access to or make unauthorized use of computer software. Typical attacks include brute force method, man in the middle, keyloggers etc. In brute force method the hacker will try to hack the software by brute force way or a dictionary of words and combinations to gain access to a computing system. Man in the middle attack occurs when a hacker eavesdrops on the network between two computers and secretly relays the information impersonating as the computers at both ends on the network. Keyloggers are malicious programs which are installed on the computers without the knowledge of the users which monitor each and every keystroke made on the system and it relays the information to the hacker.

7.5 Experimental Results and Result Analysis of SteganoDB

As proposed in Chapter 6 SteganoDB provides a new method of storing a database of related information in an image using Steganography method.

7.5.1 Experimental Results of SteganoDB

In order to test how the proposed steganoDB system will behave for various amount of data, a sample project was created using Microsoft .net technology. All the DB operations proposed as part of Chapter 6 were implemented on an image as part of these tests.

The processing of the algorithm using the software developed will involve the below steps:

First the proposed process will give an option to the user to select the cover image. The image can be of any format. However, in order to have a reduced file size, it is proposed to use the jpeg format for storing the data. According to the data
structure specified, the user can provide the information/data. This data is to be hidden in the image. This data is next encoded and stored inside the cover image which preserves the data and makes it more secure. For experimentation purpose it was chosen to insert a simple structure like given below to store basic personal details. The Data Definition for each of the record can be different as the data definition relies on the JSON structure. The data definition can be different for each record as the json structure is flexible enough to allow this.

The corresponding JSON structure for the table,

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Country</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>24</td>
<td>Germany</td>
</tr>
</tbody>
</table>

Will be as below:

```json
{
  "Name":"Lena",
  "Age":"24",
  "Country":"Germany"
}
```

For example, the next record could be like below:

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Country</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kumar</td>
<td>29</td>
<td>India</td>
<td>9847232111</td>
</tr>
</tbody>
</table>

**Input:**

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Kumar</td>
</tr>
<tr>
<td>Age</td>
<td>29</td>
</tr>
<tr>
<td>Country</td>
<td>India</td>
</tr>
<tr>
<td>Phone</td>
<td>9847232111</td>
</tr>
</tbody>
</table>
Output of SteganoDB structure:

{
"Name":"Kumar",
"Age":"29",
"Country": "India",
"Phone": "999"
}

There are separate options for inserting, upserting and deleting data. If there are no records existing, it will display an error saying that no records could be found. At the retrieval side the data is first decoded from the cover image and then this data is passed to the user. The experimentation results were done on a PC by developing a Graphical User Interface in Windows to store the data into SteganoDB and then retrieve and display it. The records as per the input will be inserted into the image in JSON format using Steganography.

Fig. 7.2 Resultant Image after inserting the given records

Unlike other database security methods, the overhead of providing security is not there as the method used is very simple. Due to the simplicity of the method this it can be also suited for modern mobile and embedded systems. Since the user can change the cover image as per his/her wish nobody suspects about the image and json text inside of it. Furthermore, the system can also be extended to be used as a client
server system. Another important thing is that only users who have the SteganoDB package will be able to insert and edit the data on the images.

### 7.5.2 Performance Analysis of SteganoDB

**Criteria 1: Comparison of Qualitative Analysis of Data**

The comparison of Data Quality was done by evaluating data storage on a normal text file against data stored using SteganoDB in an image. For the analysis the data retrieval quality and data extraction time is considered in different cases according to various uses like databases, text storage, data storage etc. The analysis is shown below:

**Table 7.2 Comparison of Data Quality**

<table>
<thead>
<tr>
<th>Data Qualities</th>
<th>Data insertion in SteganoDB</th>
<th>Data insertion in Text file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>High</td>
<td>Normal</td>
</tr>
<tr>
<td>Timeliness</td>
<td>High</td>
<td>Normal</td>
</tr>
<tr>
<td>Relevance</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Maintainability</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Security</td>
<td>Very high</td>
<td>Low</td>
</tr>
<tr>
<td>Data insertion speed</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Data retrieval speed</td>
<td>High</td>
<td>Average</td>
</tr>
<tr>
<td>Data Specification</td>
<td>Clear and Precise</td>
<td>Average</td>
</tr>
</tbody>
</table>

The tests were done to compare the insertion of records to a text file and into SteganoDB. The insertion of records into an image when compared to a normal text file was equally accurate and comparatively faster to insert. However, when you compare the maintainability aspect, from the perspective of inserting and deleting or
updating records, the SteganoDB will be much easier when compared to a normal text file. From the view point of security also SteganoDB is much better as compared to a text file. Another person will not be able to extract records without the help of a decoding program. Data insertion and retrieval speed also is as high as a text file and in some cases, it can be even higher as once the records are read, Json format is much faster to traverse when compared to the other formats of records. From the results it is apparent that the data insertion to SteganoDB is much superior with respect to ease of access and security when compared to text file insertion.

Criteria 2: Image Quality Analysis by calculating PSNR, MSE and SSIM values

The stego image quality is analyzed after inserting the records using SteganoDB structure into the cover image. The image quality metrics was calculated five times by inserting different number of records and calculated the PSNR ratio, MSE and SSIM values of (24 bit) image. The different test conditions are, adding 1 record using SteganoDB, the second time the same was done after inserting ten records. Subsequently the tests were done after inserting 20, 50 and 100 records. Testing multiple times with varied payloads will allow to identify how the image quality will be varying under various amounts of data.

Image used for the tests – Input image

Fig.7.3 Test image
### Inputs Used:

#### Table 7.3 Inputs

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Lena</td>
</tr>
<tr>
<td>Age</td>
<td>24</td>
</tr>
<tr>
<td>Country</td>
<td>Germany</td>
</tr>
<tr>
<td>Name</td>
<td>Kumar</td>
</tr>
<tr>
<td>Age</td>
<td>29</td>
</tr>
<tr>
<td>Phone</td>
<td>9847232111</td>
</tr>
<tr>
<td>Name</td>
<td>Gopal</td>
</tr>
<tr>
<td>Address</td>
<td>Kailas Nagar, Pattom</td>
</tr>
<tr>
<td>PIN</td>
<td>695004</td>
</tr>
<tr>
<td>Name</td>
<td>Rohit</td>
</tr>
<tr>
<td>Surgery</td>
<td>Elbow</td>
</tr>
<tr>
<td>Age</td>
<td>27</td>
</tr>
<tr>
<td>Blood group</td>
<td>B+ve</td>
</tr>
<tr>
<td>Train Name</td>
<td>Iland Express</td>
</tr>
<tr>
<td>Time of Arrival</td>
<td>12 O clk</td>
</tr>
<tr>
<td>Name</td>
<td>Krishana</td>
</tr>
<tr>
<td>Age</td>
<td>60</td>
</tr>
<tr>
<td>Country</td>
<td>India</td>
</tr>
<tr>
<td>Name</td>
<td>Rupa</td>
</tr>
<tr>
<td>Phone</td>
<td>9847238112</td>
</tr>
<tr>
<td>Name</td>
<td>Nirupama</td>
</tr>
<tr>
<td>Phone</td>
<td>9747236112</td>
</tr>
<tr>
<td>Author</td>
<td>A.P.J. AbdulKalam</td>
</tr>
<tr>
<td>Book Name</td>
<td>Wings of Fire</td>
</tr>
<tr>
<td>Bank</td>
<td>ICICI</td>
</tr>
<tr>
<td>Branch</td>
<td>Pulimoodu, tvm</td>
</tr>
<tr>
<td>A/c No.</td>
<td>116600723</td>
</tr>
</tbody>
</table>

Etc.....
For identification of readers each record is given different colors. For testing purpose and analysis up to 100 records were used. Above mentioned data in the table are some of the sample records that have been used. As mentioned in the above table, the format of each record can be different when it is inserted into the SteganoDB.

For comparison and analysis 5 test cases are chosen. The values of PSNR, MSE and SSIM of the resultant stego images are given below in the table.

Table 7.4 Image quality analysis of SteganoDB

<table>
<thead>
<tr>
<th>No. of records</th>
<th>Image size</th>
<th>Data size</th>
<th>SSIM</th>
<th>MSE</th>
<th>PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>768KB</td>
<td>51 bytes</td>
<td>.9997</td>
<td>.0445</td>
<td>61.6456 db</td>
</tr>
<tr>
<td>10</td>
<td>768KB</td>
<td>508 bytes</td>
<td>.9996</td>
<td>.0447</td>
<td>61.6284 db</td>
</tr>
<tr>
<td>20</td>
<td>768KB</td>
<td>1018 bytes</td>
<td>.9995</td>
<td>.0447</td>
<td>61.6258 db</td>
</tr>
<tr>
<td>50</td>
<td>768KB</td>
<td>2,548 bytes</td>
<td>.9994</td>
<td>.0447</td>
<td>61.6200 db</td>
</tr>
<tr>
<td>100</td>
<td>768KB</td>
<td>5098 bytes</td>
<td>.9993</td>
<td>.0448</td>
<td>61.6193 db</td>
</tr>
</tbody>
</table>

Fig.7.4(a) Image quality analysis using SSIM  
Fig.7.4(b) Image quality analysis using MSE
From the table and graph, it is seen that the PSNR value is varying from 66 to 67 up to 100 records. Hence it can clearly identify that the image quality is good and noise is less. From these data it can be assumed that nearly 2000 records can be embedded into a similar sized image by maintaining its PSNR value not less than 35 or 30. It has been observed that the SSIM value is close to 1 that means the image is almost similar to the original image also. The MSE value also varies only very little even after increasing the number of records.

**Criteria 3: Image Quality Analysis by Histogram Analysis**

For making sure about the image quality analysis 5 test cases are considered. The image was analyzed before inserting the data using histogram analysis also the other cases are, the records inserted in the order 1, 10, 20, 50 and 100 in the cover image and analyzed using histogram of the corresponding stegoimages. The Changes can be clearly visible through the RGB histograms. The variation of pixels can be easily found out; hence even minute changes can be reflected in histogram analysis. The cover images, the stego images of the test cases and its corresponding histograms are provided below for analyzing image quality in detail.
Fig. 7.5 (a1) Original Image
Test 1 – After inserting 1 record

Fig. 7.5 (b1) Histogram of Original Image

Fig. 7.5 (a2) Test case 1
Test 2 – After inserting 10 records

Fig. 7.5 (b2) Histogram of Test case 1

Fig. 7.5(a3) Test case 2
Test 3 – After inserting 20 records

Fig. 7.5(b3) Histogram of Test case 2

Fig. 7.5 (a4) Test case 3
Test 4 – After inserting 50 records

Fig. 7.5 (b4) Histogram of Test case 3

Fig. 7.5 (a5) Test case 4
Test 5 – After inserting 100 records

Fig. 7.5 (b5) Histogram of Test case 4

Fig. 7.5 (a6) Test case 5

Fig. 7.5 (b6) Histogram of Test case 5

Fig. 7.5 Resultant images and its histogram after inserting data
From the histogram analysis also it is clearly identified that the difference of original image and embedded images is quite less. After embedding 1 record the histogram varies very little bit only. Similarly, the differences after embedding 10, 20, 50 and 100 record is also not much different. The difference is so minimal that on comparing very carefully, one can find the difference between resultant images.

Advantage and disadvantages

1. More secure than maintaining data on text files

2. Does not have overhead of large database systems

3. Strict data definition is not needed.

4. Non related data can be stored.

5. Can be implemented easily on any type of computing platform (mobile, tablet etc.)

6. Maintenance of data is also easy.

7. Disadvantage – if more records are inserted there will be image degradation. Hence capacity is limited.

7.5.3 Discussion on SteganoDB

From the test results it can be seen that SteganoDB provides an easy to use and secure alternative to store data especially in embedded systems where there is a requirement to store limited amount of data but with complete security. This thesis presents a new novel and non-conventional architecture to create a secure SteganoDB which will be storing related data in JSON format. This helps the application developers to include a security mechanism within their applications to store the data securely. Unlike other database security methods, the overhead of providing security is not there as the method used is very simple. Due to the simplicity of the method this can be also suited for modern mobile and embedded systems. Since the user can
change the cover image as per his/her wish, nobody suspects about the image and stegan inside of it. Furthermore, the system can also be extended to be used as a client server system. Another important thing is that only users who have the SteganoDB package will be able to insert and edit the data on the images. One drawback of the system is that because data is stored on to images, there will certainly be limitation in storing data before the image quality starts degrading. To overcome this as a future expansion the system can be adapted to store the data into video and audio files which will allow us to store much more data.

7.6 Experimental Results and Result Analysis of Pixel Pattern Based Steganography

To overcome the shortcomings of regular steganography algorithms, a new algorithm for steganography has been proposed, which is a pixel pattern based steganography and involves hiding the message within an image by using the properties of RGB values of the image. Whenever possible at pixel level or with minimum changes to the pixel wherever it is not possible. The details have been discussed in Chapter 6.

7.6.1 Experimental Results of Pixel Pattern Based Steganography

To demonstrate the algorithm, a small application was developed in .net. The first step is to encode the text to be hidden into the image which needs to be used. The application allows for both embedding and extracting the text to be hidden and show the o/p to the user. For testing purpose 24-bit image is used.
### Input

**Data**

{“Perfection is achieved, not when there is nothing more to add, but when there is nothing left to take away.”}

---

**Fig. 7.6 Given Image**

To embed the data into the image it fetches the data and the cover image. Before embedding the data, the AES encryption algorithm encrypts the data and it will be embedded within the image using Pixel Pattern based Steganography algorithm. The algorithm scans all the pixel positions and find out the most appropriate pixel positions. The pixel positions are stored for revoke the data. The data is encoded and embed into the given image Fig. 7.6 using Pixel Pattern based Steganography and the resultant stego image Fig. 7.7 is created.

### Output

**Fig. 7.7 Stego Image**

The pixel position information is stored along with metadata of the image which holds the information about pixels.
Fig. 7.8. Screen shots of Information is stored in metadata of image

In this Fig. 7.8 the comment field have been used to store the encrypted pixel location information. But it can be any field in the metadata or the algorithm can also be adapted to use multiple fields in the metadata. For extracting the text, the extraction algorithm is invoked which will select a stegano image and allow to extract the message and also display result on the screen. The output of the extraction algorithm is given below.

**Output:**

“Perfection is achieved, not when there is nothing more to add, but when there is nothing left to take away.”

For extracting the data, the data extraction algorithm is executed. First of all, it fetches the stegano image and find out its metadata from properties comments Fig. 7.8. From the metadata it finds out the pixel positions and decode the message
and then AES decryption algorithm is executed and resolve the data in the correct form as output.

7.6.2 Performance Evaluation of Pixel Pattern Based Steganography

To check whether the method proposed in this thesis is superior to the other methods, it is compared against some of the other existing steganography methods. For comparing PSNR ratio, MSE values and SSI values are calculated. PSNR value of the encoded image (Peak Signal to Noise Ratio) is normally calculated to find the noise of the image. Higher the PSNR ratio specifies that the image keeps good quality. The minimum value of MSE denotes that both the given images are having almost same quality.

Higher the PSNR value means the reconstruction of image is of higher quality. For all type of color images with RGB values per pixel the definition of PSNR is the same except the MSE is the sum over all squared value differences divided by image size and by three. At the same time, for colour images the image is converted to a different colour space and PSNR is reported against each channel of that colour space. The insertion capacity of image can be tested using the PSNR ratio.

Criteria 1: Checking the image size of the cover image and stego image after inserting a specified bit of data in various methods.

For the experiment, the standard colour images (24 bit) of $512 \times 512$ (786,432 byte) have been used and the inserted information is text message. In each image same amount of data is inserted and checked. Different sized images are used to test for each Steganography methods like LSB, MSB, RGB and the proposed steganography. The change of the cover image is observed and the details are furnished in the Table 7.3.
Table 7.5 Comparison of Image size and Stego image

<table>
<thead>
<tr>
<th>Technology Used</th>
<th>Cover Image</th>
<th>Stego Image</th>
<th>Size of cover Image</th>
<th>Size of stego Image</th>
<th>Data Inserted</th>
<th>Size of Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSB Steganography</td>
<td>![Image]</td>
<td>![Image]</td>
<td>37.5 KB</td>
<td>147 KB</td>
<td>India</td>
<td>40 bits</td>
</tr>
<tr>
<td>MSB Steganography</td>
<td>![Image]</td>
<td>![Image]</td>
<td>29.1 KB</td>
<td>31.0 KB</td>
<td>India</td>
<td>40 bits</td>
</tr>
<tr>
<td>RGB steganography</td>
<td>![Image]</td>
<td>![Image]</td>
<td>193 KB</td>
<td>257 KB</td>
<td>India</td>
<td>40 bits</td>
</tr>
<tr>
<td>The Proposed Algorithm</td>
<td>![Image]</td>
<td>![Image]</td>
<td>432 KB</td>
<td>434 KB</td>
<td>India</td>
<td>40 bits</td>
</tr>
<tr>
<td>The Proposed Algorithm</td>
<td>![Image]</td>
<td>![Image]</td>
<td>881 KB</td>
<td>882 KB</td>
<td>India</td>
<td>40 bits</td>
</tr>
<tr>
<td>The Proposed Algorithm with modification to pixels</td>
<td>![Image]</td>
<td>![Image]</td>
<td>57 KB</td>
<td>60 KB</td>
<td>India</td>
<td>40 bits</td>
</tr>
</tbody>
</table>

Table 7.5 shows that when comparing with the original image, the stego image increases in size for all the methods but comparatively the new proposed method does not increase much. Because the size is lesser for the new proposed algorithm, image quality also does not degrade at all. The SSIM, MSE, PSNR values also will not change much due to this. Hence the proposed method is better in this aspect when compared against pure LSB method MSB and RGB.
Criteria 2: Image quality analysis by checking PSNR, MSE and SSIM

For going further evaluation in detail, here calculating the value of PSNR, SSIM and MSE values in different sized images by inserting the same size of data.

Table 7.6 Image quality analysis using variable sized images

<table>
<thead>
<tr>
<th>Technology Used</th>
<th>Size of image</th>
<th>Size of Data</th>
<th>SSIM</th>
<th>MSE</th>
<th>PSNR Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSB Steganography</td>
<td>37.5 KB</td>
<td>40 bits</td>
<td>.9554</td>
<td>9.6613</td>
<td>38.2805 dB</td>
</tr>
<tr>
<td>LSB Steganography</td>
<td>190 KB</td>
<td>40 bits</td>
<td>.9941</td>
<td>2.5076</td>
<td>44.1395 dB</td>
</tr>
<tr>
<td>MSB Steganography</td>
<td>29.1 KB</td>
<td>40 bits</td>
<td>.9947</td>
<td>1.8663</td>
<td>45.4205 dB</td>
</tr>
<tr>
<td>MSB Steganography</td>
<td>190 KB</td>
<td>40 bits</td>
<td>.9941</td>
<td>2.6766</td>
<td>53.3972 dB</td>
</tr>
<tr>
<td>RGB Steganography</td>
<td>193 KB</td>
<td>40 bits</td>
<td>.9531</td>
<td>2.2704</td>
<td>44.112 dB</td>
</tr>
<tr>
<td>RGB Steganography</td>
<td>432 KB</td>
<td>40 bits</td>
<td>.9964</td>
<td>5.8890</td>
<td>49.9728 dB</td>
</tr>
<tr>
<td>Proposed Algorithm</td>
<td>432 KB</td>
<td>40 bits</td>
<td>1.00</td>
<td>0.00</td>
<td>1.#INF</td>
</tr>
<tr>
<td>Proposed Algorithm</td>
<td>881 KB</td>
<td>40 bits</td>
<td>1.00</td>
<td>0.00</td>
<td>1.#INF</td>
</tr>
<tr>
<td>Proposed Alg. with pixel modification</td>
<td>57KB</td>
<td>40 bits</td>
<td>.9999</td>
<td>0.00733</td>
<td>79.0206 dB</td>
</tr>
<tr>
<td>Proposed Alg. with pixel modification</td>
<td>432 KB</td>
<td>40 bits</td>
<td>1.00</td>
<td>.000429</td>
<td>91.3480 dB</td>
</tr>
</tbody>
</table>

The above Table 7.6 uses the same image took for the preparation of Table 7.5.

From the above test results, it is apparent that the results will show variance in values due to difference in the image size. Also when the image size becomes less, the ability of data storing decreases. Hence its PSNR value is decreasing. To avoid this and maintain the better quality of stego image, it is always better to use an image which is
larger in size for storing the data. For storing large amount data, it is better to use large sized cover image.

Over LSB, MSB and RGB steganography, the proposed algorithm gives good results than other steganography techniques. Also it can be understood that while taking good sized image gives good perfection. In LSB, MSB and RGB steganography if the size of the image is reduced the quality of the image decreases, which means that for storing large data, large sized image should be used to keep image quality. But in the case of proposed method it is not needed in all cases. Commonly it will keep all the pixels in the same as original image, only in rare cases changes the pixels. Hence it keeps the image quality.

Criteria 3: Image quality analysis by inserting data with different sizes

To make sure the performance of the proposed algorithm, further tests are conducted with different size of data embedding in fixed sized image to clearly observe what changes will happen in the PSNR value and its image quality. For the second set of tests the image of Tajmahal (24 bit) was used but the size of inserted text increased first to 17, 34, 50, 100 and 150 bytes. The size of the image used was 128 KB.

Input Image

Fig. 7.9 Tajmahal (Cover image)
Table 7.7 Image quality analysis using variable size of data

<table>
<thead>
<tr>
<th>Technology Used</th>
<th>Size of Data</th>
<th>SSIM</th>
<th>MSE</th>
<th>PSNR Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSB</td>
<td>17 bytes</td>
<td>.99993</td>
<td>0.00336</td>
<td>82.41242 dB</td>
</tr>
<tr>
<td>MSB</td>
<td>17 bytes</td>
<td>.99999</td>
<td>0.00199</td>
<td>84.69039 dB</td>
</tr>
<tr>
<td>RGB</td>
<td>17 bytes</td>
<td>1</td>
<td>0.00187</td>
<td>84.94745 dB</td>
</tr>
<tr>
<td>Proposed Algorithm</td>
<td>17 bytes</td>
<td>1.00</td>
<td>0.00</td>
<td>#INF</td>
</tr>
<tr>
<td>LSB</td>
<td>34 bytes</td>
<td>.99998</td>
<td>0.00605</td>
<td>79.85313 dB</td>
</tr>
<tr>
<td>MSB</td>
<td>34 bytes</td>
<td>.99999</td>
<td>0.00363</td>
<td>82.07162 dB</td>
</tr>
<tr>
<td>RGB</td>
<td>34 bytes</td>
<td>1</td>
<td>0.00324</td>
<td>82.56271 dB</td>
</tr>
<tr>
<td>Proposed Algorithm</td>
<td>34 bytes</td>
<td>1</td>
<td>0.00</td>
<td>1.#INF</td>
</tr>
<tr>
<td>LSB</td>
<td>50 bytes</td>
<td>.99996</td>
<td>0.01147</td>
<td>77.07855 dB</td>
</tr>
<tr>
<td>MSB</td>
<td>50 bytes</td>
<td>.99999</td>
<td>0.00546</td>
<td>80.30161 dB</td>
</tr>
<tr>
<td>RGB</td>
<td>50 bytes</td>
<td>1</td>
<td>0.00548</td>
<td>80.28348 dB</td>
</tr>
<tr>
<td>Proposed Algorithm</td>
<td>50 bytes</td>
<td>1</td>
<td>0.00</td>
<td>1.#INF</td>
</tr>
<tr>
<td>LSB</td>
<td>100 bytes</td>
<td>.99992</td>
<td>0.02131</td>
<td>74.38677 dB</td>
</tr>
<tr>
<td>MSB</td>
<td>100 bytes</td>
<td>.9995</td>
<td>0.01002</td>
<td>77.66094 dB</td>
</tr>
<tr>
<td>RGB</td>
<td>100 bytes</td>
<td>.99999</td>
<td>0.01026</td>
<td>77.56313 dB</td>
</tr>
<tr>
<td>Proposed Algorithm</td>
<td>100 bytes</td>
<td>1.00</td>
<td>0.00</td>
<td>1.#INF</td>
</tr>
<tr>
<td>LSB</td>
<td>150 bytes</td>
<td>.99999</td>
<td>0.03031</td>
<td>72.85688 dB</td>
</tr>
<tr>
<td>MSB</td>
<td>150 bytes</td>
<td>.99999</td>
<td>0.01409</td>
<td>76.18274 dB</td>
</tr>
<tr>
<td>RGB</td>
<td>150 bytes</td>
<td>.99999</td>
<td>0.01388</td>
<td>76.24656 dB</td>
</tr>
<tr>
<td>Proposed Algorithm</td>
<td>150 bytes</td>
<td>1.00</td>
<td>0.00</td>
<td>1.#INF</td>
</tr>
</tbody>
</table>
From the above test results, it can be found that the proposed algorithm has better PSNR ratio and maintain the image quality compared with other techniques.
Since the MSE, SSIM and PSNR values show no change, it can be understood that it doesn’t change the pixels in the image when compared with the original image. Hence there is no degradation in the image quality.

Criteria 4: Calculating the Data insertion and Extraction time

For comparing the efficiency of proposed algorithm the data insertion and data extraction time of various techniques were calculated. For that three size of data is considered then inserted into the same image (Fig. 7.10) and corresponding values are checked. The results are shown in the below table 7.6.

Table 7.8. Comparison of insertion extraction time for different steganography techniques

<table>
<thead>
<tr>
<th></th>
<th>LSB Steganography</th>
<th>MSB Steganography</th>
<th>RGB Steganography</th>
<th>Pixel Pattern based Steganography</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size of data</td>
<td>17bytes</td>
<td>17bytes</td>
<td>17bytes</td>
<td>17bytes</td>
</tr>
<tr>
<td>Time for Embedding</td>
<td>500 millisecs</td>
<td>580 millisecs</td>
<td>550 millisecs</td>
<td>2000 millisecs</td>
</tr>
<tr>
<td>Time for Extraction</td>
<td>800 millisecs</td>
<td>800 millisecs</td>
<td>850 millisecs</td>
<td>700 millisecs</td>
</tr>
<tr>
<td>Size of data</td>
<td>34 bytes</td>
<td>34 bytes</td>
<td>34 bytes</td>
<td>34 bytes</td>
</tr>
<tr>
<td>Time for Embedding</td>
<td>600 milli secs</td>
<td>600 milli secs</td>
<td>600 millisecs</td>
<td>2050 millisec</td>
</tr>
<tr>
<td>Time for Extraction</td>
<td>900 millisecs</td>
<td>910 millisecs</td>
<td>850 millisecs</td>
<td>750 millisecs</td>
</tr>
<tr>
<td>Size of data</td>
<td>50 bytes</td>
<td>50 bytes</td>
<td>50 bytes</td>
<td>50 bytes</td>
</tr>
<tr>
<td>Time for Embedding</td>
<td>900 millisecs</td>
<td>900 millisecs</td>
<td>950 millisecs</td>
<td>2100 millisecs</td>
</tr>
<tr>
<td>Time for Extraction</td>
<td>1100 millisecs</td>
<td>1250 millisec</td>
<td>1050 millisec</td>
<td>900 millisecs</td>
</tr>
<tr>
<td>Size of data</td>
<td>100 bytes</td>
<td>100 bytes</td>
<td>100 bytes</td>
<td>100 bytes</td>
</tr>
<tr>
<td>Time for Embedding</td>
<td>1000 milli secs</td>
<td>1020 milli secs</td>
<td>1000 milli secs</td>
<td>2100 milliscs</td>
</tr>
<tr>
<td>Time for Extraction</td>
<td>1300 millisecs</td>
<td>1300 milli secs</td>
<td>1250 milli secs</td>
<td>950 millisecs</td>
</tr>
<tr>
<td>Size of data</td>
<td>200 bytes</td>
<td>200 bytes</td>
<td>200 bytes</td>
<td>200 bytes</td>
</tr>
<tr>
<td>Time for Embedding</td>
<td>1100 millisecs</td>
<td>1200 millisecs</td>
<td>1150 millisecs</td>
<td>2100 millisecs</td>
</tr>
<tr>
<td>Time for Extraction</td>
<td>1450 millisecs</td>
<td>1500 millisecs</td>
<td>1400 millisecs</td>
<td>1000 millisecs</td>
</tr>
</tbody>
</table>
From the Table 7.8 it is seen that LSB, MSB and RGB steganography technique’s data insertion time and extraction time is proportional to the data inserted. If the size of the data is increased the data insertion time is also increased, the same is happened in the case of data extraction too. One more thing is noted that, in all other cases the data extraction time is more than the data insertion time. But in the case of proposed algorithm the data insertion time is more compared to other algorithm. That is one of the limitations, but this data insertion time is not varies according to the size of data inserted. In the case of small size of data only it is an issue, if in the case of
large amount of data, (more than 300 bytes of data) the proposed algorithm took less
time compared to the other algorithms. Also one more advantage is, it took less time
for extracting the embedded text. That means its data retrieval time is faster than other
algorithms, it is having good response. That makes things easier, the hacker may not
think about an algorithm is running to retrieve the data. This makes the data safer.

**Criteria 5: Quality analysis of data**

The quality analysis of data is done on the basis of the result of data retrieval
time from Table 7.8, availability of data in time, completeness of data and correctness
of data. And also considered the steganalysis of these techniques. The tests were done
in different parameters and in different conditions. According to the results it is
concluded in the below table.

<table>
<thead>
<tr>
<th>Techniques</th>
<th>Reliability of data</th>
<th>Availability of data</th>
<th>hackable</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSB Steganography</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
</tr>
<tr>
<td>MSB Steganography</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
</tr>
<tr>
<td>RGB Steganography</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
</tr>
<tr>
<td>Pixel Pattern based</td>
<td></td>
<td></td>
<td>Very difficult</td>
</tr>
<tr>
<td>Steganography</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

From the comparison it is noted that the reliability and availability of pixel
pattern based steganography algorithm is comparable to the other steganography
algorithms however the main difference comes in terms of security. The new
proposed algorithm is very difficult to hack as it does not change any of the pixels in
the image and also uses a custom encryption to store the data in the metadata. Even in
case slight modification in pixels is done, it will be very difficult to detect and cannot
be re-engineered even if detected. Hence this algorithm is much superior than the traditional steganography algorithms.

The highlights of pixel pattern based steganography algorithm is,

1) The image is virtually not changed. Hence there is no visible difference at all.
2) The image size will increase slightly to store the additional metadata.
3) It can be easily shared using any method.
4) Noise is less compared to any other technique.
5) Comparatively large amount of data can be stored without changing picture quality.
6) Nobody can guess about any type of attacks inside the image.
7) More secure than any other RGB methods.

7.6.3 Discussion on Pixel Pattern Based Steganography

Pixel pattern based steganography algorithm presents an improved steganography method for embedding secret message bit in image metadata fields based on the RGB values and the position of the pixels. The image pixels will be changed only for characters where the algorithm cannot find a pixel which can represent it. Since only the metadata is modified, the stego image looks exactly the same as original image or it will be very difficult to identify the changes for the human eye. Only the size of the stego image will increase slightly however in our test cases this has been found to be comparable with other steganography methods. This research was aimed towards the development of a new and improved data hiding technique based on RGB based steganography without changing the image. Some of the possible application areas include transmitting small secret messages, using an
image as a password token by encrypting and hiding password using this technique, simply adding a hidden signature to an image etc.

7.7 Experimental Results and Result Analysis of Mobile Data Protection

7.7.1 Experimental Results of Mobile Application Protection

In order to test the methods discussed in this section, a windows mobile based application was developed with additional steganography [19] based protection included. The development environment used was Visual Studio 2013 and tests were run on two mobiles. Mobile-1 was a Lumia 535 running Windows Phone version 8.1 and Mobile-2 was a Lumia 822 running Windows phone 8.1. Both phones have 1GB RAM. The software can however be run on even mobile phones having 512 MB RAM. Since most modern mobile phones nowadays come with minimum of 1GB of RAM the software will run fine on any mobile phone. For steganography the application needs to have image manipulation capabilities.

The sample application that was developed for the testing purpose is a very simple application, to track expenses based on location where the expense occurred. The application was run when it was protected using Steganography and also when it was not protected. The application which was protected was run on the Lumia 822 and the unprotected one was run on Lumia 535 respectively. During the registration process the user information is gathered and it sends to the server via message. The collected information is act as the key to the application. The output of the authentication part gives one of the input of the protection algorithm.
**Input:**

Name: Arun Kumar

Email: arun.kumar@outlook.com

Phone: 9895312209

Address: 124#Lane 15, T-Nagar, Chennai-30

Registration Serial no:
U2FsdGVkX19L7/IiYCBYbur74I5oNTBL/nBaMPfgg+s=

**Output of SteganoDB Algorithm:**

```json
{
  "First Name": "Arun",
  "Last Name": "Kumar",
  "Email": "arun.kumar@outlook.com",
  "Address Line1": "124#Lane 15",
  "Address Line2": "T-Nagar",
  "Address Line3": "Chennai-30",
  "Phone": "9895312209"
  "Key": "U2FsdGVkX19L7/IiYCBYbur74I5oNTBL/nBaMPfgg+s="
}
```

**Stego Image Token Creation**

The input data is gathered while registering the software. These information is applied on SteganoDB algorithm and the structure is created. It is encoded using AES algorithm. This will be embedding within the cover image given using Pixel Pattern based Steganography data embedding algorithm. Then the resultant Stego image is created.
Input data:
{
  "First Name":"Arun",
  "Last Name":"Kumar",
  "Email":"arun.kumar@outlook.com",
  "Address Line1":"124#Lane 15",
  "Address Line2":"T-Nagar",
  "Address Line3":"Chennai-30",
  "Phone":"9895312209"
  "Key":"U2FsdGVkX19L7/IiYCBYbur74I5oNTBL/nBaMPf9g+s="
}

Output Image (Stego image Created):

This stego image acts as the protection token of the application. It is unique hence if the application is pirating with the stego image also the application will not run. Hence this token protects the application from piracy.

Now consider the various execution conditions of the applications,

Test case 1 – Application was first developed without including the protection module and then, it was tried to execute on both the mobiles. It can be opened in any mobile when copied and installed.
This test case proves that if a mobile app is not protected, it can be easily copied and started on any mobile easily.

**Test case 2** – Application which is protected with the stego method. When the application is initially run, it checks for the stego image, matches the values between the image and the mobile properties and if both matches then the application is started.
When the mobile protection module is enabled in the app, it will execute only on the mobile on which the app is registered.

**Test case 3** – In case the application is pirated and installed on a different mobile, the application will check for the registration information and will show an error message if the application is not registered. It will also provide an option to go to the registration screen and register the app.

Fig. 7.17 Protected application gives error when trying to run on mobile on which it has not been registered

Basically here the idea is that when an unauthorized user tries to run the app rather than giving error out the app will provide an option for him to register the app by paying and become a legitimate user. This will allow the app developer to get more users as well.
7.7.2 Experimental Results of file/message Protection

As the part of experimentation for file protection using steganography, a sample application was created which will accept an image and file/message to protect as input and produce the o/p file. This file was send to the receiving side, where the deciphering application exists then decrypt the file and show the output to the receiver.

The Authentication information is got from the user becomes one of the input of file/message protection. It is the key to deciphering the message. The algorithm will automatically detect the IMEI number which can be used for generating the encryption key. The next input is the file/message to protect. Using this algorithm data can be protected from text files. It can also be enhanced to get data from document files and pdfs also. When the receiver receives the image using the same algorithm, it can decipher the message using the key.

**Input**

<table>
<thead>
<tr>
<th>Inputs (key)</th>
<th>Cover Image</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{ &quot;First Name&quot;: &quot;Arun&quot;, &quot;Last Name&quot;: &quot;Kumar&quot;, &quot;Email&quot;: &quot;arun.kumar@outlook.com&quot;, &quot;Phone&quot;: &quot;9895312209&quot; &quot;Key&quot;: &quot;U2FsdGVkX19L7/IiYCBYbur74I5oNTBL/nBampfgg+s=&quot;</code></td>
<td>Fig. 7.18 Cover image</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input (Message)</th>
<th>Fig. 7.19 Stego image token</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Come and collect your needed stuffs as soon as possible.”</td>
<td></td>
</tr>
</tbody>
</table>

Output (Stego Token)
After sending the image to the receiver, receives the picture as message which will become the input at the receiving side and data extraction algorithm of Pixel Pattern based Steganography extracts the data from the stego image token Fig. 7.19 and it will be decoded using AES decryption algorithm.

At the receiving side when the authenticated user opens the message/file it can be automatically extracted and the contents are decrypted from the stegano image then shown to the receiver. If an unauthorized person tries to access this file/message this cannot be resolved and it looks like an innocent image. Hence the content can be preserved. By using this algorithm confidential messages or passwords can be protected and can send to the user safely. Also this algorithm can be used to protect confidential files/ databases or some other data safely inside mobiles.

**Output**

“Come and collect your needed stuffs as soon as possible.”

The same method can be used for protecting of text from files like pdfs and documents also.

### 7.7.3 Performance Evaluation of Mobile Data Protection

As a part of the performance analysis the stegano image was created with the user information email, unique information IMEI number of the phone and address information. For this example, the IMEI number of the phone will be used to uniquely identify it. For testing this three different cover images of 24 bit were used.
Input:

![Fig. 7.20 Cover image 1](image1.png) ![Fig. 7.21 Cover image 2](image2.png) ![Fig. 7.22 Cover image 3](image3.png)

The above cover images were used to embed data and produce the stego image. These images were specifically selected because they look quite innocent similar to normal logo at the same time provide a decent enough color range.

The registration information stored is as below which will be encrypted value of IMEI number along with the other registration info created after the processing of identity algorithm.

**Input:**

```json
{
  "First Name": "Arun",
  "Last Name": "Kumar",
  "Email": "arun.kumar@outlook.com",
  "Address Line1": "124#Lane 15",
  "Address Line2": "T-Nagar",
  "Address Line3": "Chennai-30",
  "Phone": "9895312209"
  "Key": "U2FsdGVkX19L7/IiYCBYbur74I5oNTBL/nBaMPfgg+s="}
```

**Output (Stego images)**

![Fig. 7.23 Stego image 3](image3.png) ![Fig. 7.24 Stego image 3](image3.png) ![Fig. 7.25 Stego image 3](image3.png)
Criteria 1: Comparison of image quality of the resultant image

In this mobile application protection algorithm, pixel pattern based steganography methodology is used. In section 7.6.2 the detail image quality analysis was done using pixel pattern based algorithm for the various criteria. Hence here the image quality analysis is done only by checking the input used for protecting data in different cover images and by checking its PSNR, MSE and SSIM ratio.

Table 7.10 Image quality analysis of different output images

<table>
<thead>
<tr>
<th></th>
<th>Size of Image</th>
<th>Size of Data</th>
<th>Output</th>
<th>PSNR</th>
<th>SSIM</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cover image 1</td>
<td>1.22 MB</td>
<td>243 bytes</td>
<td>Stego image 1</td>
<td>95.1352 db</td>
<td>.99999</td>
<td>.00014</td>
</tr>
<tr>
<td>Cover image 2</td>
<td>756 KB</td>
<td>243 bytes</td>
<td>Stego image 2</td>
<td>94.3465 db</td>
<td>.99999</td>
<td>.00021</td>
</tr>
<tr>
<td>Cover image 3</td>
<td>598 KB</td>
<td>243 bytes</td>
<td>Stego image 3</td>
<td>91.1281 db</td>
<td>.99999</td>
<td>.00045</td>
</tr>
</tbody>
</table>

In all these cases it can be seen that the image quality has not degraded much. Hence any hacker trying to analyze data, it is not at all easy. Because of the variation of image is minor, the case of suspicion on the image is less. From the detail study of image quality analysis of section 7.6.2 it is obviously known that noise is quite less when compared to other techniques.

Criteria 2: File/Message as input and checking the image quality

Considering the scenario of protection of message/file, the size of data could be much more than the scenario of application protection. Due to this reason, for comparing image quality here 3 text files are taken. It is having more size. Those files are embedded inside the cover image 3. After this the Stegoimage’s PSNR, MSE and SSIM values were calculated for analysis.
Table 7.11 File protection’s image quality analysis

<table>
<thead>
<tr>
<th></th>
<th>Size of Image</th>
<th>Size of Data</th>
<th>PSNR</th>
<th>SSIM</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cover image3</td>
<td>598 KB</td>
<td>500 bytes</td>
<td>90.7660 dB</td>
<td>.99999</td>
<td>.00049</td>
</tr>
<tr>
<td>Cover image3</td>
<td>598 KB</td>
<td>4,895 bytes</td>
<td>90.4317 dB</td>
<td>.99999</td>
<td>.00053</td>
</tr>
<tr>
<td>Cover image3</td>
<td>598 KB</td>
<td>19,586 bytes</td>
<td>87.9556 dB</td>
<td>.99999</td>
<td>.00094</td>
</tr>
</tbody>
</table>

This Table 7.11 shows the same conclusion of the previous table’s results as once again. It can be seen that more amount of data can be protected safely inside the same image and it is safer than any other existing steganography technique.

Criteria 3: Quality Analysis of Data in File/Message/Application Protection

The conditions considered in this criteria are easiness in data embedding, accessibility of data in extraction time, completeness of data in retrieval, availability of data in time, data secureness and easy storage and retrieval. These conditions are compared for different data’s and different size of data. According to the results data quality specification of data is summarized in the table below:

Table 7.12 Quality analysis of file/message/ Application Protection

<table>
<thead>
<tr>
<th>Data Quality</th>
<th>File Protection</th>
<th>Message Protection</th>
<th>Application Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>High</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Availability</td>
<td>Fast Retrieval</td>
<td>Fast Retrieval</td>
<td>Fast Retrieval</td>
</tr>
<tr>
<td>Security</td>
<td>High</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Maintenance</td>
<td>Easy</td>
<td>Easy</td>
<td>Easy</td>
</tr>
<tr>
<td>Data Specification</td>
<td>Clear</td>
<td>Clear</td>
<td>Clear</td>
</tr>
</tbody>
</table>
The data extraction in this technique (Pixel Pattern based Steganography) is faster than compared to other techniques. It has proven in the section 7.6.2 criteria 4. In all these protections data quality is good and it has enough security due to the support of steganography and cryptography techniques. Since the algorithm using SteganoDB structure the data insertion, deletion or any type of operation done easily if needed. It is easy to maintain and use, compared to any other techniques.

Criteria 4: Security Analysis - Comparison with Other Protection Techniques

The security of this protection framework is compared by analyzing against some of the known protection techniques. The considerations of comparison are based on excellency in the security, implementation cost, easiness to maintain in the usage and ability to resist the hacking possibilities. After the analysis the results are summarized as follows.

Table 7.13 Comparison of proposed protection from traditional protection techniques

<table>
<thead>
<tr>
<th>Technique</th>
<th>Security level</th>
<th>Cost benefit</th>
<th>Protection against hacking &amp; piracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial numbers</td>
<td>Low</td>
<td>Low</td>
<td>Very low. Easily replicable</td>
</tr>
<tr>
<td>Passwords</td>
<td>Low</td>
<td>High</td>
<td>Easily hackable</td>
</tr>
<tr>
<td>Cryptography</td>
<td>Average</td>
<td>Average</td>
<td>Can be hacked based on brute force methods</td>
</tr>
<tr>
<td>Cryptography + micro controller</td>
<td>High</td>
<td>Low</td>
<td>High cost of implementation</td>
</tr>
<tr>
<td>Hardware dongles</td>
<td>Cannot be used with mobiles</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Steganography + h/w identification + encryption</td>
<td>Very high</td>
<td>Very high</td>
<td>Extremely high protection.</td>
</tr>
</tbody>
</table>

Some of the mobile apps which use traditional method of protection just uses a serial number for protecting it against pirating. However, the new method proposed uses the hardware features like IMEI number or mac address which are unique to a
mobile. Hence it offers better protection against the traditional methods. Secondly when compared to desktop protection methods, things like a hardware dongle cannot be attached against a mobile. The proposed method uses a combination of steganography and encryption to protect the hardware generated key against any hacking attempt. This also makes the proposed method quite secure when compared against traditional protection methods.

Criteria 5: How this protection algorithm tackles threats

As part of the performance analysis for measuring security of this protection framework we can consider the processing cycles of the threats, data gathering way by analyzing how this algorithm exceeds. Piracy methods can be simple, medium or can involve complex activities.

Simple methods of piracy – sharing serial number, copying across the software to other computers

If someone tries to soft lift the program from one system to another, it will not work since during the start itself the software will check for the hdd /mac serial number and compare it against the stored one in the stegano image.

Distributing the application via internet or getting license for one mobile and trying to install on other mobiles also will not allow hackers access as each of the installed instance is unique and will work on only one system on which it has been installed.

Methods of piracy with medium complexity – Reverse Engineering using Steganalysis and Brute force password cracking.
A steganalyst will use various techniques during steganalysis to find the steganography method used. Then he will try to reverse engineer the same method and get back the text that is stored within the image. This includes trying to find out if any pixel in the image has been changed by histogram analysis and trying to find again which bit of the pixel has been changed once a match is got. Steganalysis will fail as the images do not have any variation as such.

The second method is for a hacker to use brute force method to crack the encrypted text. Even if a hacker tries to decrypt the text it will be near impossible to do this. Finally, copying the software to another machine will not work as the hardware checks will fail.

Even using a super computer will take years of processing to reverse engineer an encrypted text using AES algorithm. Hence the time required for this will make it virtually ineffective in using it as an attacking method.

**Complex methods of piracy** – Try to use packet sniffing methods on the network.

Suppose someone tries to use packet sniffing tools to get the registration info, then all the hacker will get is some bits of the image which he will not be able to do reverse engineer back to original form. Hence this method of software registration cannot be hacked.

Advantages of the application protection using Steganography and combination of h/w features are:
1. This method is virtually non-recognizable to a hacker since it uses both steganography and cryptography methods to make the protection of mobile app more efficient and secure.

2. Fairly easy to implement without need of any additional expensive hardware.

3. Internal working of the registration process is completely non-transparent to the user.

4. Does not use any additional memory or h/w to implement.

5. This method of protection will work by reusing the internet connection to authenticate the application.

7.7.4 Discussion on Mobile Data Protection

This section discussed the performance of protection of data in mobiles using steganography methods. Considering the time needed for the attacks makes it practically impossible to crack/hack. From the results it is seen that this method of protection is much better than the existing methods and helps to protect the mobile apps against piracy. This same method can be extended to protect mobile based banking, Ecom applications etc. when critical information like password or other information is transmitted via insecure network like internet.

7.8 Experimental Results and Result Analysis of Digital Data Protection Framework

In this particular protection framework for software protection, the proposed method uses pixel pattern based steganography method as well as the AES encryption method also for identification it makes use of hardware features. These methods have been discussed in the earlier Chapters 3, 4 and 5. The Digital Data Protection framework using these methods is proposed in Chapter 6.
7.8.1 Experimental Results of Software Protection

In order to test the protection capabilities, a standard windows application was created. The sample application first creates the stegano image token and then packaged along with the software. When the application starts to execute, it will read the stegano image and will decrypt the hidden text. In the next step it will compare the encrypted text with the hardware properties like the hdd serial or MAC address. If these value match, then the software will execute. Otherwise the software will refuse to execute.

To show the experimental results of the algorithm a prototype was created. It works in four steps.

User side:

1. At the user end there will be a registration wizard as well as the main software. The registration software will get the HDD serial number, encrypt the HDD serial number and send it via email to the authentication server on the internet.

2. Along with the main software there is a module present that will check the presence of the key file, it will decode the key file and will check whether the decoded value is matching with the actual HDD key. If both of them matches, then only the software will execute.

3. There is an update module that will regularly check the license internet server for any software updates by passing along the registration info as well. If the registration info is not valid, it will exit and will make the software unusable.

Server side:

1. At the server side there is a separate software module that will receive the registration request along with the encrypted HDD serial number. It will hide
the HDD serial number within the image and will send the image back to the user to complete the registration.

The AES algorithm for decoding text from key image will be at the user side. At the server side there will be encoding of the text into the key image.

**Authentication Algorithm Result**

The registration wizard is the first step while the protected software is executed. This will accept the buyer’s address and other user information. The unique hardware key and this information will be sent to the license issuing authority (server). To the user it will simply appear as a random generated key. However, in actual it’s the HDD serial number that gets send in an encrypted fashion. Alternatively, other unique hardware features like MAC address, motherboard serial number etc. can be used as unique values.

The Registration wizard will create a text file with registration info and send it to the registration server. The text file got after executing the authentication algorithm will be similar to the one shown below:

**Output:**

```
{
    "FirstName":"Arun",
    "LastName":"Kumar",
    "Email":"arun.kumar@outlook.com",
    "AddressLine1":"12#1Lane 15",
    "AddressLine2":"T-Nagar",
    "AddressLine3":"Chennai-30",
    "Phone":"9895122269",
    "Key":"U2FsdGVkX19L7/IlYCBYbur741SoNT8L/nBMPf gg+s=
}
```

Fig. 7.26 Authentication Information
Key File Creation

This algorithm is executed from the server side on receiving the registration key from the user. The information got from the user via text file is embed in the cover image using SteganoDB algorithm and Pixel Pattern based Steganography algorithm. Then hide the key into an image and this resultant image file is send to the user.

The creation of stego-image formed is below Fig. 7.27. All image formats like .bmp, .jpg etc. can be applicable as the cover image. Since LSB based steganography is using normal standard RGB scheme can be used. And it is very simple to make use of also.

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>&quot;First Name&quot;:&quot;Arun&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;Last Name&quot;:&quot;Kumar&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;Email&quot;:&quot;<a href="mailto:arun.kumar@outlook.com">arun.kumar@outlook.com</a>&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;Address Line1&quot;:&quot;124#Lane 15&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;Address Line2&quot;:&quot;T-Nagar&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;Address Line3&quot;:&quot;Chennai-30&quot;,</td>
<td></td>
</tr>
<tr>
<td>&quot;Phone&quot;:&quot;9895312209&quot;</td>
<td></td>
</tr>
<tr>
<td>&quot;Key&quot;:&quot;U2FsdGVkX19L7/IiYCBYbur74I5oNTBL/nBaMPfgg+s=&quot;</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 7.27 Lena.jpg file is used as cover image

Input

Output

{                                          |
| "First Name":"Arun",                       |
| "Last Name":"Kumar",                       |
| "Email":"arun.kumar@outlook.com",          |
| "Address Line1":"124#Lane 15",             |
| "Address Line2":"T-Nagar",                 |
| "Address Line3":"Chennai-30",              |
| "Phone":"9895312209"                       |
| "Key":"U2FsdGVkX19L7/IiYCBYbur74I5oNTBL/nBaMPfgg+s=" | |

Fig. 7.28 cat.bmp file is used as cover image
The key is embedded inside the original image and produces the stego-image as shown in the Fig. 7.27, Fig. 7.28. Human eyes perception capability will not be able to identify the embedded key within the image. Hence the picture can safely hide the key inside it with maximum confidentiality.

**Software Execution**

The user on receiving the key file can use it to register the software. When the software is registered, it will open as intended. Subsequently each time when the software is started, it will get the HDD key and compare the HDD key with the steganoDB if both of them matches, then only will the execution be allowed. Else if the comparison fails the execution will stop with an error message.

**Output:**

![Image of registered software execution]

Fig. 7.29 Screen shots of Execution of sample software

If it does not match, then the software will not execute. It will give the message of registering the software properly.

**Output:**

![Image of unregistered software execution]

Fig. 7.30 Screen shots of failure execution.
7.8.2 Experimental Results of file Protection

As part of the experimentation for file protection using steganography, a sample application was created which will accept a file to protect. It will accept input as file and embed the data into a cover image, then produce the o/p as an image file. This file can be stored/ sent. Whenever needed these data can be retrieve using the data retrieval algorithm. The text can be extracted from file formats like word documents or pdfs using the standard functions provided by the Operating Systems.

For getting the text from a word document, provides api’s which can be used. Once the input file is provided to the API, it will read the document and extract the text from the document. The Authentication information becomes one of the input of file protection. It is the key to deciphering the message. The algorithm will automatically detect the HDD serial / mac address which can be used for generating the encryption key. The next input is the file to protect.

Input:

```
{  
"First Name":"Arun",  
"Last Name":"Kumar",  
"Email":"arun.kumar@outlook.com",  
"Phone":"9895312209"  
"Key":"U2FsdGVkX19L7/IiYCBYbur74I5oNTBL/nBaMPfgg+s="}  
```

Fig. 7.31 Input File

```
Alice was beginning to get very tired of having nothing to do, as she had peeped into the book her sister was reading, but it had no pictures or conversations in it, and what is the use of a book, thought Alice, without pictures or conversations?

As she was considering in her own mind (as well as she could, for the hot day made her feel very sleepy and stupid), whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.
```

Input (Cover Image)

Fig. 7.32 Cover Image
The contents of the word file get using API functions and embed it inside the cover image using Pixel Pattern based Steganography algorithm. Also the authentication information is embed using the Steganodb structure kept inside the cover image (24 bit). And gives the output as picture.

Output: (Stego image)

![Stego image](image)

This can be keep within the system or it can be sent to whom you like to exchange the file/information. At the receiving side when the authenticated user can open the file can be extracted the information for use. If an unauthorized person tries to access this file, it cannot be resolved and it looks like an innocent image. Hence the content can be preserved. By using this algorithm confidential messages, databases or passwords can be protecting and can send to the user safely.

While extracting the contents of the file it will check the authentication information, if the authentication is right the contents of the file is extracted and gives the output as text format.
7.8.3 Performance Evaluation of Digital Data Protection

Digital data protection involves the protection of software as well as data files using the steganography method is proposed. The initial step is the registration part which gives the registration information as below:

**Authentication Output:**

```
{
"First Name":"Arun",
"Last Name":"Kumar",
"Email":"arun.kumar@outlook.com",
"Address Line1":"124#Lane 15",
"Address Line2":"T-Nagar",
"Address Line3":"Chennai-30",
"Phone":"9956128119"
"Key":"vel5uWYBk970H4DXoAOseGx90JTXMFWL4Dbzlc63lx5adBmTcg3x0r0lrzhEBKx"
}
```

This registration information was stored into a cover image Fig. 7.37 by using the Pixel Pattern based Steganography Algorithm proposed in Chapter 6. Once this is done, for each run of the application which is protected by the algorithm will decrypt the data stored in the image and compare it with the hardware properties. If they match the software is allowed to run.

**Cover Image:**

Fig. 7.35 Cover image logo

216
As part of the performance analysis the cover image used is Fig. 7.37. Stegano image was generated with the user information and the encrypted HDD serial and inserted it into an image. The registration information stored is as below, which will be encrypted value of HDD serial number or MAC address. For test purpose 4 test cases of data were used. Two for HDD and two for MAC address were taken.

**Input data’s used for test:**

**HDD 1:**

{3WXpf6UXRV5nxAleQml9BRohIKw0Whs/XKGww4OdPb0yJ8qNZc0nSeSM5Kxldosp}

**HDD 2:**

{hAJN64oQ1j0=}

**MAC Address 1:**

{WZrk6Sbl6d0IQTxOC0FE0pZEI52OzX7}

**MAC Address 2:**

{4rTvk7zBIC31+FM/ONmUGlVa4xu0oYTT}

{  
"First Name":"Arun",  
"Last Name":"Kumar",  
"Email":"arun.kumar@outlook.com",  
"Address Line1":"124#Lane 15",  
"Address Line2":"T-Nagar",  
"Address Line3":"Chennai-30",  
"Phone":"9956128119"  
"Key":"3WXpf6UXRV5nxAleQml9BRohIKw0Whs/XKGww4OdPb0yJ8qNZc0nSeSM5Kxldosp"
}

In the same way the other inputs are changed according to the new hdd2, Mac Address 1, Mac Address 2 etc.
Criteria 1 - Comparison of image quality by embedding the given inputs

The image quality is measured by calculating the value of PSNR, MSE and SSIM values. Different possible inputs are embedded into the given cover image and calculated the corresponding stego image’s PSNR, SSIM and MSE values.

Table 7.14 Image quality of stego image used for software protection

<table>
<thead>
<tr>
<th>Size of Image</th>
<th>Size of Data</th>
<th>PSNR (db)</th>
<th>SSIM</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>250k</td>
<td>218 bytes (HDD2)</td>
<td>99.7989 dB</td>
<td>.99999</td>
<td>.00047</td>
</tr>
<tr>
<td>250k</td>
<td>238 bytes (MAC Address 1)</td>
<td>98.8298 dB</td>
<td>.99999</td>
<td>.00059</td>
</tr>
<tr>
<td>250k</td>
<td>238 bytes (MAC Address 2)</td>
<td>98.8298 dB</td>
<td>.99999</td>
<td>.00059</td>
</tr>
<tr>
<td>250k</td>
<td>270 bytes (HDD1)</td>
<td>94.4382 dB</td>
<td>.99999</td>
<td>.00021</td>
</tr>
</tbody>
</table>

From the Table 7.14 it is clearly seen that the image quality of the stego image is good. Hence the noise is less. If adding more data as checking constraint, it is maintaining the image quality. This is an added advantage that if enhance the protection by giving other information this algorithm gives good image quality and good support for the protection.

Performance of the steganography algorithm and data insertion algorithm are done detail in the sections 7.6.2. In the case of the digital data protection the same algorithm was repeating here. Hence the data insertion and extraction work as in the similar pattern. For making sure about the data quality and data retrieval capability is good, data quality checks are done again in this algorithm again.
Criteria 2 - Comparison of Quality analysis of Data for the above mentioned input’s

The comparison of data quality with other techniques are done detail in section 7.6.2 in criteria 4 and criteria 5. Here the data insertion and data extraction time from the stego image is calculated for various size of data, to analyze whether the data can be extracted in time, it is correct and reliable. Tests were done using the given inputs and results were obtained clearly and completely in time. It is comparatively fast retrieval than any other technique. The results are summarized as below:

Table 7.15 Data quality analysis of software protection

<table>
<thead>
<tr>
<th>Size of Image</th>
<th>Size of Data</th>
<th>Data insertion time</th>
<th>Data extraction time</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>250k</td>
<td>270 bytes (HDD1)</td>
<td>1.5 sec</td>
<td>800 milli secs</td>
<td>100%</td>
</tr>
<tr>
<td>250k</td>
<td>218 bytes (HDD2)</td>
<td>1 sec</td>
<td>750 milli sec</td>
<td>100%</td>
</tr>
<tr>
<td>250k</td>
<td>238 bytes (MAC Address 1)</td>
<td>1 sec</td>
<td>750 milli secs</td>
<td>100%</td>
</tr>
<tr>
<td>250k</td>
<td>238 bytes (MAC Address 2)</td>
<td>1 sec</td>
<td>750 milli secs</td>
<td>100%</td>
</tr>
</tbody>
</table>

The Table 7.15 shows that it is having the output similar to the section 7.6.2. These values shows that the data retrieval is fast when compared to any other algorithm. Since the data retrieval is fast the other processes will be fast. As a result of this, the response time will be good. Due to this the totality of the algorithm gets good performance and good response time. One more advantage of this is the whole process is not transparent to the user. It runs in background and hence suspicion to a hacker can be avoided. This makes the protection better. Since the data is unique to a system, it will always remain accurate. While retrieving also, the algorithm will always store and retrieve the same data without any data corruptions.
Criteria 3 - Comparison of Image quality analysis of Data for file protection

The same method can be used for storing of data from files like text, pdf into image and securely transfer it to the receiver. The data inserted in the case of file protection algorithm is comparatively more. Hence once again the image quality analysis was done in that case. Here as the encryption key the hdd serial number or mac address will be used for identification/authentication of user. As a one-time process initially this information would have been already passed from the sender to the receiver. Image quality is analyzed after embedding the variable size of data from the file to the cover image, then the stego image’s PSNR, SSIM and MSE values are calculated. The results are shown below.

Table 7.16 Image quality of stegoimage used for file protection

<table>
<thead>
<tr>
<th>Cover image1</th>
<th>Size of Image</th>
<th>Size of Data</th>
<th>PSNR</th>
<th>SSIM</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>250k</td>
<td>1000 bytes</td>
<td>94.4382 dB</td>
<td>.99999</td>
<td>.00021</td>
<td></td>
</tr>
<tr>
<td>250k</td>
<td>1500 bytes</td>
<td>91.55028 dB</td>
<td>.99999</td>
<td>.00041</td>
<td></td>
</tr>
<tr>
<td>250k</td>
<td>2000 bytes</td>
<td>88.9292 dB</td>
<td>.99999</td>
<td>.00075</td>
<td></td>
</tr>
</tbody>
</table>

From the Table 7.16 it is clear that if more data is embedded into the image also will change the pixels in the image minimally. Since similar pixel values position is kept as meta data and those pixels which are changed will have only very minor change in the pixel values, hence it gives least changes in the resultant image. This will maintain the image quality and gives better protection for data. Due to this reason even 2000 bytes of data embedded in the image got good PSNR ratio and SSIM values. Noise is less compared to any other technique.

Criteria 4 - Comparison with Other Protection Techniques

Since it is a protection frame work on which the testing is more concentrated as part of the security analysis. This proposed protection technique is compared with
other protection techniques used, usually for protecting software. Normally for protection purposes we are using cryptography, steganography, dongles, serial numbers and other protection techniques. It’s usability, convenience, implementation easiness, maintenance of algorithm, extensibility, cost and its security capability is considered. The result of the analysis given in precise below.

Table 7.17 Comparison of digital data protection from other protection techniques

<table>
<thead>
<tr>
<th>Technique</th>
<th>Security level</th>
<th>Cost</th>
<th>Protection against hacking &amp; piracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial numbers</td>
<td>Low</td>
<td>Low</td>
<td>Very low. Easily replicable</td>
</tr>
<tr>
<td>Hardware dongles</td>
<td>Medium</td>
<td>Very High</td>
<td>High protection</td>
</tr>
<tr>
<td>Cryptography</td>
<td>High</td>
<td>Low</td>
<td>Medium protection</td>
</tr>
<tr>
<td>Steganography</td>
<td>High</td>
<td>Low</td>
<td>Medium protection</td>
</tr>
<tr>
<td>Steganography + hardware features + encryption</td>
<td>Very high</td>
<td>Very Low</td>
<td>Extremely high protection.</td>
</tr>
</tbody>
</table>

The protection technique is used as a combination of cryptography, steganography and identification features which gives more protection than any one of the protection methods using alone. Cryptography has limitations to protecting data. All types of data cannot be protect using cryptography alone. There is chance of getting data reverse engineered using bruteforce attacks. So we can’t surely protect the data using cryptography. Steganography also sometimes is prone to steganalysis methods, to unhide the data. Hardware dongles mostly give strong protection, but it is costly and also it can’t be used for all types of systems.

Criteria 5 – Security Analysis: How will this protection method protect against hacking.

Usually several threats can influence the algorithm and destruct the protection used. This thesis considered the hacking methods and analyzed its methods on
processing and extraction of data. The various piracy methods are Simple, medium piracy and complex piracy.

**Simple methods of piracy** – sharing serial number, copying across the software to other computers. If someone tries to soft lift the program from one system to another, it will not work since during the start itself the software will check for the HDD /mac serial number and compare it against the stored one in the stegano image.

Distributing the application via internet or getting license for one computer and trying to install on other computers also will not allow hackers to access, as each of the installed instance is unique and will work on only one system on which it has been installed.

**Methods of piracy with medium complexity** – Reverse Engineering using Steganalysis and Brute force password cracking. A steganalist will use various techniques during steganalysis to find the steganography method used. One of the method is anybody will try to reverse engineering, and get back the text that is stored within the image. Another one is, trying to find out change of pixels by histogram analysis and trying to find again which bit of the pixel has been changed once a match is got. Steganalysis will fail as the images do not have any variation as such.

The second method is for a hacker to use brute force method to crack the encrypted text, even if a hacker tries to decrypt the text it will be near impossible to do this. Finally copying the software to another machine will not work as the hardware checks will fail. Even using a super computer will take years of processing to reverse engineer an encrypted text using AES algorithm. Hence the time required for this will make it virtually ineffective in using it as an attacking method.
**Complex methods of piracy** – Try to use packet sniffing methods on the network which will analyze the packets being send and then alter them for using the software.

By considering this how the proposed algorithm exceeds the hacking methods and threats is narrated here.

### Table 7.18 Threats and its recovery capability of digital data protection

<table>
<thead>
<tr>
<th>Hacking Methods</th>
<th>Recovery</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steganalysis</td>
<td>Since the whole data is not kept inside the image steganalysis will fail. Since the pixels are not changed much, it is difficult to find variation of pixel. It is also very difficult to retrieve data, due to double encryption provided and usage of the metadata concept.</td>
</tr>
<tr>
<td>Crypt analysis</td>
<td>Double encryption levels are provided because of this it is difficult to retrieve the data completely.</td>
</tr>
<tr>
<td>Brute force</td>
<td>Brute force method to crack the software will not work as first will have to reverse engineer the steganographic method which is not possible and next will have to identify the AES key used which is also extremely difficult. Hence this protection is virtually un crackable.</td>
</tr>
<tr>
<td>Password cracking</td>
<td>Brute force method to crack the software will not work as first will have to reverse engineer the steganographic method which is not possible and next will have to identify the AES key used which is also extremely difficult. Hence this protection is virtually un crackable.</td>
</tr>
<tr>
<td>Copying of software</td>
<td>If the software is tries to copy, it can be copy to another system. But it will not execute in another system.</td>
</tr>
<tr>
<td>Soft lifting</td>
<td>During each run the software will check for the HDD /mac serial number and compare it against the stored data in the stegano image. Hence it can’t run on the soft lifted system.</td>
</tr>
<tr>
<td>Renting</td>
<td>Renting software to another person will not work as the software is tied to a particular system via hardware properties. Hence copying and execution to another system is not possible.</td>
</tr>
<tr>
<td>Packet sniffing</td>
<td>Someone tries to use packet sniffing tools to get the registration info, then all the hacker will get is some bits of the image which he will not be able to reverse engineering back to original form.</td>
</tr>
<tr>
<td>Distributing the application via internet</td>
<td>If the application tries to spread through net because of the first level checking, it fails. It will not survive.</td>
</tr>
</tbody>
</table>

Some of the various hacking methods used include steganalysis, brute force password cracking method, copying the software to another machine etc. From the...
above table it is clearly understood that this protection framework can exceed hacking and cracking to an extent.

The advantages of this method of software protection are

1. The s/w is tied to the uniqueness of computer like the hdd serial number or mac address. Hence the s/w will not work if it is pirated from another.

2. Since the registration process uses strong AES and Steganography to hide the key, hackers will not be able to access the data. Since Steganography algorithm used is the new pixel value based algorithm which does not change the values of the pixels in an image, it results an unhackable system.

2. Process is simple and can be integrated into any type of software easily. The registration process is quite simple to implement.

3. Since the hardware feature uses the hard disk unique serial number it can be applied to all normal systems.

4. No such extra h/w piece like dongle or some other h/w is used and hence this method is cost effective.

**7.8.4 Discussion on Digital Data Protection Framework**

Pixel pattern based steganography algorithm presents an improved steganography method for embedding secret message bit in image metadata fields based on the RGB values and the position of the pixels. In this section pixel pattern based steganography along with hardware unique properties and encryption were used for software protection as well as protection of file. Since there is triple level protection was being used, it is almost impossible for a hacker to identify and hack.
From the results it is clearly seen that this method can be used effectively for protection of data.