The International Journal of Multi-Disciplinary Research
ISSN: 3471-7102

Forensics Tool – Digital Image Steganography in Java

(Conference ID: CFP/107/2017)

Author: Mr. SIKOTA SIKOTA
namataa592@gmail.com

BSc Information Security and Computer Forensics

Information and Communications University

ABSTRACT

People have desired to keep certain sensitive communications secret for thousands of years. In our new age of digital media and internet communications, this need often seems even more pressing. This paper presents general information about steganography, the art of data hiding.

The paper provides an overview of steganography, general forms of steganography, specific stenographic methods, and recent developments in the field. The information presented in this paper is also applied to a program developed by the author, and some sample runs of the program are presented.
A Digital Image Steganography Tool, a Java project will be one of the applications to address information security from misuse and hacking.

KEY WORDS:
Steganography, Steganalysis, data hiding, data security, data embedding, stego objects, watermarking, secret communications, secret messages, hidden messages, hidden channels, LSB alterations

INTRODUCTION AND BACK GROUND

In the current trends of the world, the technologies have advanced so much that most of the individuals prefer using the internet as the primary medium to transfer data from one end to another across the world. There are many possible ways to transmit data using the internet: via e-mails, chats, etc. the data transition is made very simple, fast and accurate using the internet. However, one of the main problems with sending data can be stolen or hacked in many ways. Therefore, it becomes very important to take data security into consideration, as it is one of the most essential factors that need attention during the process of data transferring.

Data security basically means protection of data from unauthorized users of hackers and providing high security to prevent data modification. This area of data security has gained more attention over the recent period of time due to the massive increase in data transfer rate over the internet. In order to improve the security features in data transfers over the internet, many techniques have been developed like: Cryptography, Steganography and digital watermarking. While Cryptography is a method to conceal information by encrypting it to 'cipher texts' and transmitting it to the intended receiver using unknown key, Steganography provides further security by hiding the cipher text into a seemingly invisible image or other formats.

According to Johnson et al., (2001), "Steganography is the art of hiding and transmitting data through apparently innocuous carriers to conceal the existence of data". The level of visibility is decreased using many hiding techniques in 'Image Modelling 'like LSB 'Manipulation', 'Masking and filtering'.

These techniques are performed by different stenographic algorithms like F5, LSB, JSteg etc. and the act of detecting the information hidden through these algorithms is called Steganalysis. "Cryptography" is the art of science used to achieve security by encoding the data to transform them into non-readable formats so that unauthorized users cannot gain access to it.

The encoded text is known as 'Cipher text' and this technique is known as encryption and this process is reversed with authorized access using the decryption technique, in which the encoded data is decoded into readable format (Kahate,2008).

'Steganography' and 'Cryptography' are closely related constructs. The hidden or embedded image, audio or a video files act as carriers to send the private messages to the destination without any security breach. Steganography techniques can be implemented on various file formats such as audio ('.mp3', '.wmv.'.etc.), video ('.mpeg', '.dat', etc.) and

images ('.jpeg', '.bmp',etc.). However, the images are the most preferred file format for this technique. At present, there are a lot of algorithms that help in executing the steganography software. These tools are (Krenn, 2004). "Digital watermarking" is described as one of the possibilities to close the gap between copyright issues and digital distribution of data. It is mainly based on Steganography techniques and enables useful safely mechanisms (Jeffrey, 2008).

It acts as a very good medium for copyright issues as it embeds a symbol or a logo in the form of a watermark, which cannot be altered manually. One critical factor to be kept in mind when using steganography is to prevent any further alterations to the originality of the image after embedding the data. Whenever the image with the secret data is transmitted over the internet and unauthorized parties may want to hack the data hidden over the image.

So, if the originality of the image has been changed then it will be easier to hack the information by unauthorized persons. In order to improve the security, the Digital watermarks are predominantly inserted as transformed digital signal into the source data using key based embedding algorithm and pseudo noise pattern.

This technique has also found big use in the notorious hands of terrorists and the September 2001

Twin tower attacks of the USA are predominantly associated with the communications using steganography. The Steganalysis aims at discovering and decrypting the suspected data transferred with the use of the available algorithms.

MATERIALS

Software and Hardware requirements:

Hardware:

Processor	Intel(R) Pentium(R) iv CPU 2.66 GHz or above
RAM	2 GB RAM or More

104 keys keyboard

Display capable of showing 65,000 colors or more

Mouse with minimum two buttons

CD – ROM Drive for installing the package.

Operating System	Windows 8.1, Linux, Solaris

Software:

Front End	ECLIPSE
	Java (jdk 1.4.1 and above)

METHODS

Related Work

Hiding data is the process of embedding information into digital content without causing perceptual degradation [1]. In data hiding, three famous techniques can be used. They are watermarking, steganography and cryptography. Steganography is defined as covering writing in Greek. It includes any process that deals with data or information within other data. According to Lou et al. [2], steganography is hiding the existence of a message by hiding information into various carriers. The major intent is to prevent the detection of hidden information.

Research in steganography technique has been done back in ancient Greek where during that time the ancient Greek practice of tattooing a secret message on the shaved head of a messenger, and letting his hair grow back before sending him through enemy territory where the latency of this communications system was measured in months [3]. The most famous method of traditional steganography technique around 440 B.C. is marking the document with invisible secret ink, like the juice of a lemon to hide information. Another method is to mark selected characters within a document by pinholes and to generate a pattern or signature [3]. However, the majority of the development and use of computerized steganography only occurred in year 2000 [4]. The main advantage of steganography algorithm is because of its simple security mechanism. Because the steganographic message is integrated invisibly and covered inside other harmless sources, it is very difficult to detect the message without knowing the existence and the appropriate encoding scheme [5]. There are several steganography techniques used for hiding data such as batch steganography, permutation steganography, least significant bits (LSB), bit-plane complexity segmentation (BPCS) and chaos based spread spectrum image steganography (CSSIS).

Research in hiding data inside image using steganography technique has been done by many researchers, for example in [6-10]. Warkentin et al. [6] proposed an approach to hide data inside the audiovisual files. In their steganography algorithm, to hide data, the secret content has to be hidden in a cover message. El-Emam [7], on the other hand, proposed a steganography algorithm to hide a large amount of data with high security. His steganography algorithm is based on hiding a large amount of data (image, audio, text) file inside a colour bitmap (bmp) image. In his research, the image will be filtered and segmented where bits replacement is used on the appropriate pixels.

These pixels are selected randomly rather than sequentially. Chen et al. [8] modified a method used in [9] using the side match method. They concentrated on hiding the data in the edge portions of the image. Wu et al. [10], on the other hand, used pixel-value differencing by partitioning the original image into non-overlapping blocks of two consecutive pixels.

This research uses a similar concept introduced by El-Emam [7]. A bitmap (bmp) image will be used to hide the data. Data will be embedded inside the image using the pixels. Then the pixels of stego image can then be accessed back in order to retrieve back the hidden data inside the image. Two stages are involved. The first stage is to come up with a new steganography algorithm in order

to hide the data inside the image and the second stage is to come up with a decryption algorithm using data retrieving method in order to retrieve the hidden data that is hided within the stego image.

DESIGN Project
Specifications

Digital image steganography tools are designed from several programming languages. This project was done in java language for the development of the Digital Image Steganography Tool. After research on the required codes to build a Digital image Steganography tool, the Digital image Steganography tool source codes were put together and edited in Notepad++ and Eclipse IDE before compiling and building different files into one Jar files. The Digital image Steganography tool is able to perform the basic encryption and decryption of the message.

Project Building Steps

The Digital Image Steganography java tool was developed in the Eclipse Development kit using researched source codes for the Tool and encryption / decryption operations, user interface and platform independent.
In this project there are two modules or steps, namely

“Creating stegano Medium”

“Getting secret information from stegano medium”

Each of the modules is described in detail as follows:

In making stegano medium side the secret information is hidden within an image file. Before hiding, for security, user has to enter a user code and secret information. A secret code will be generated using user code + secret information and this secret code will be used by the receiver to extract the secret information.
After generating secret code stegano medium will be generated. This stegano medium is the final output and expected output from the sender side. In getting secret information from stegano medium side, actually anyone may get this stegano medium that is picture with secret information, but only the person who knows secret code can read the message. Inputs for breaking the stegano medium are stegano medium and secret code.

sign in Detail

Algorithm:

Step 1: Start the process

Step 2: Enter the Secret Information

Step 3: Enter the User Code

Step 4: Load a multimedia data, here it is an Image

Step 5: Creation of Secret Code by using user + secret information

Step 6: Hiding secret information with its security into the multimedia data

Step 7: A message box showing the secret key will appear

Step 8: stop the process

Extracting secret information from Steganography medium:

Step 1: Start the process

Step 2: Enter the Secret Code

Step 3: Enter the Secret Code

Step 4: Extract secret information from stegano medium by using secret code

Step 5: Stop the Process

Software modeling

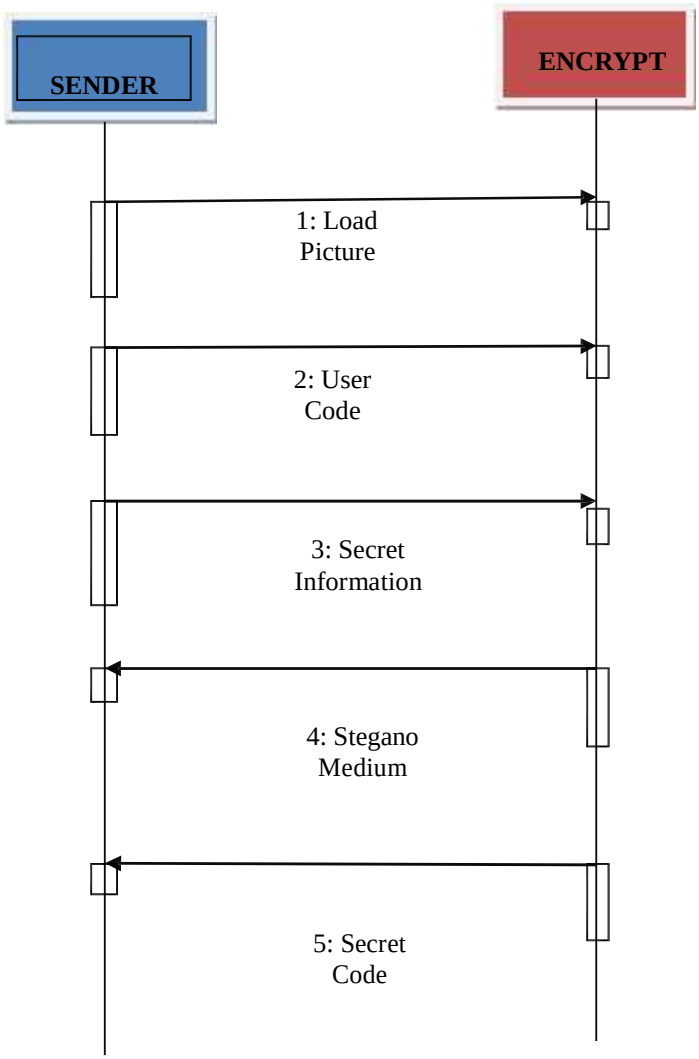


Fig: (i)

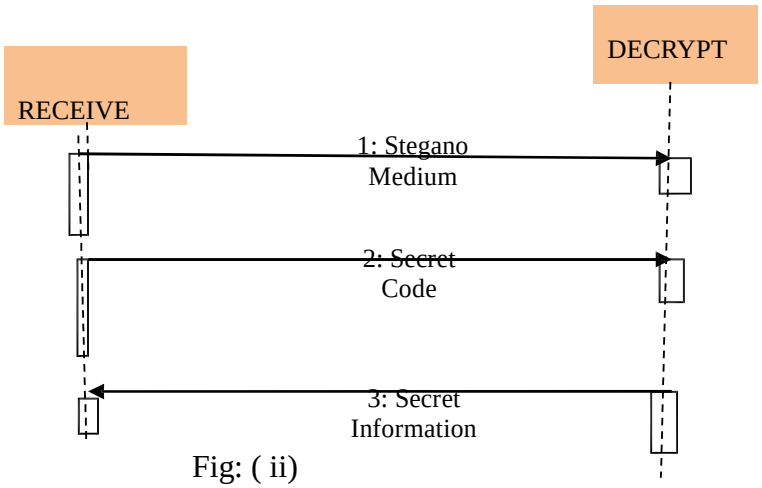
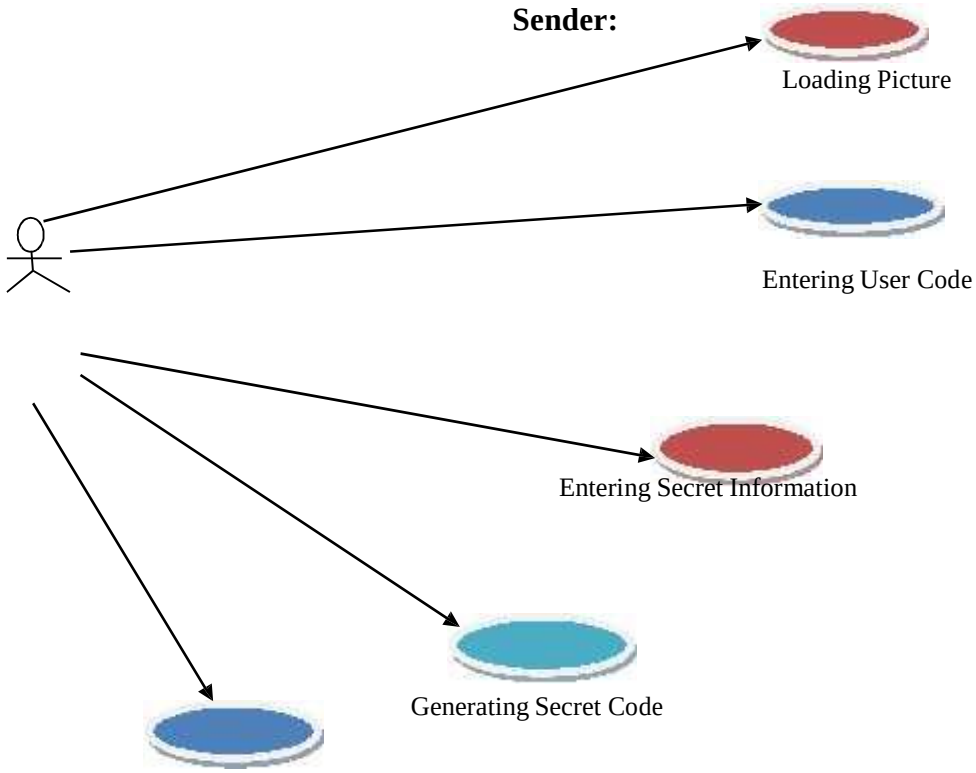


Fig: (ii)

Use Case Diagram



nzndbchiuhuu

Creating Stegano Medium

Fig: (iii)

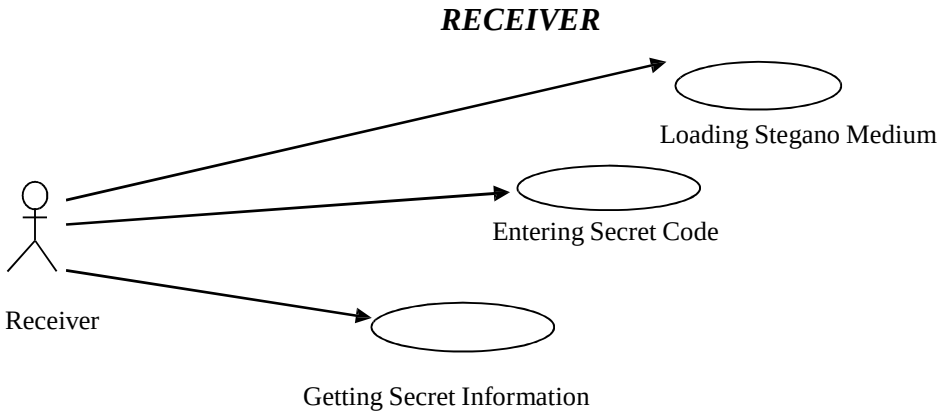


Fig: (iv)

Class Diagram

Client

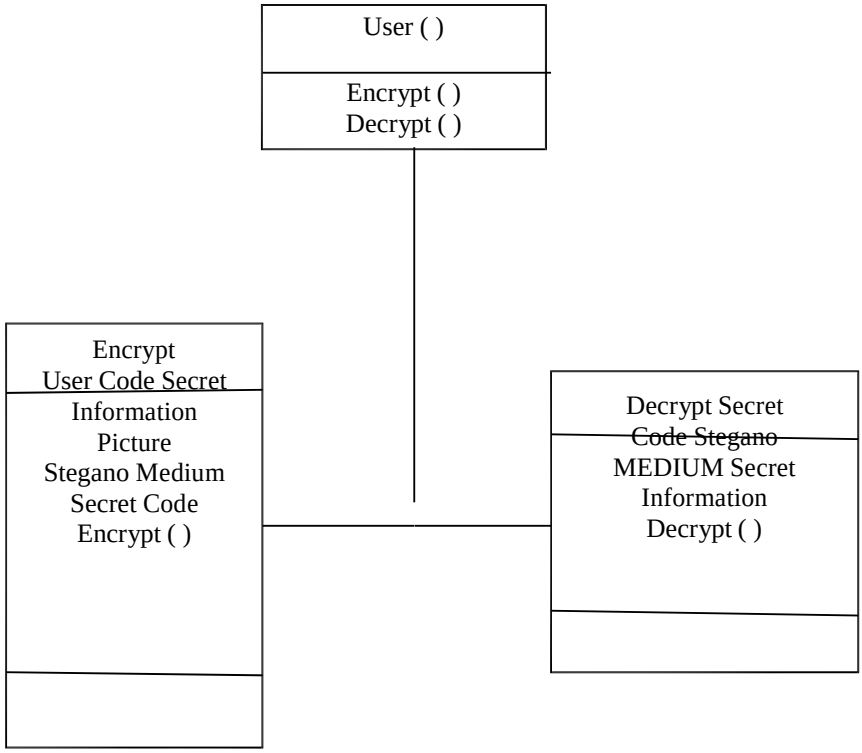


Fig: (v)

Flow Chart Diagram

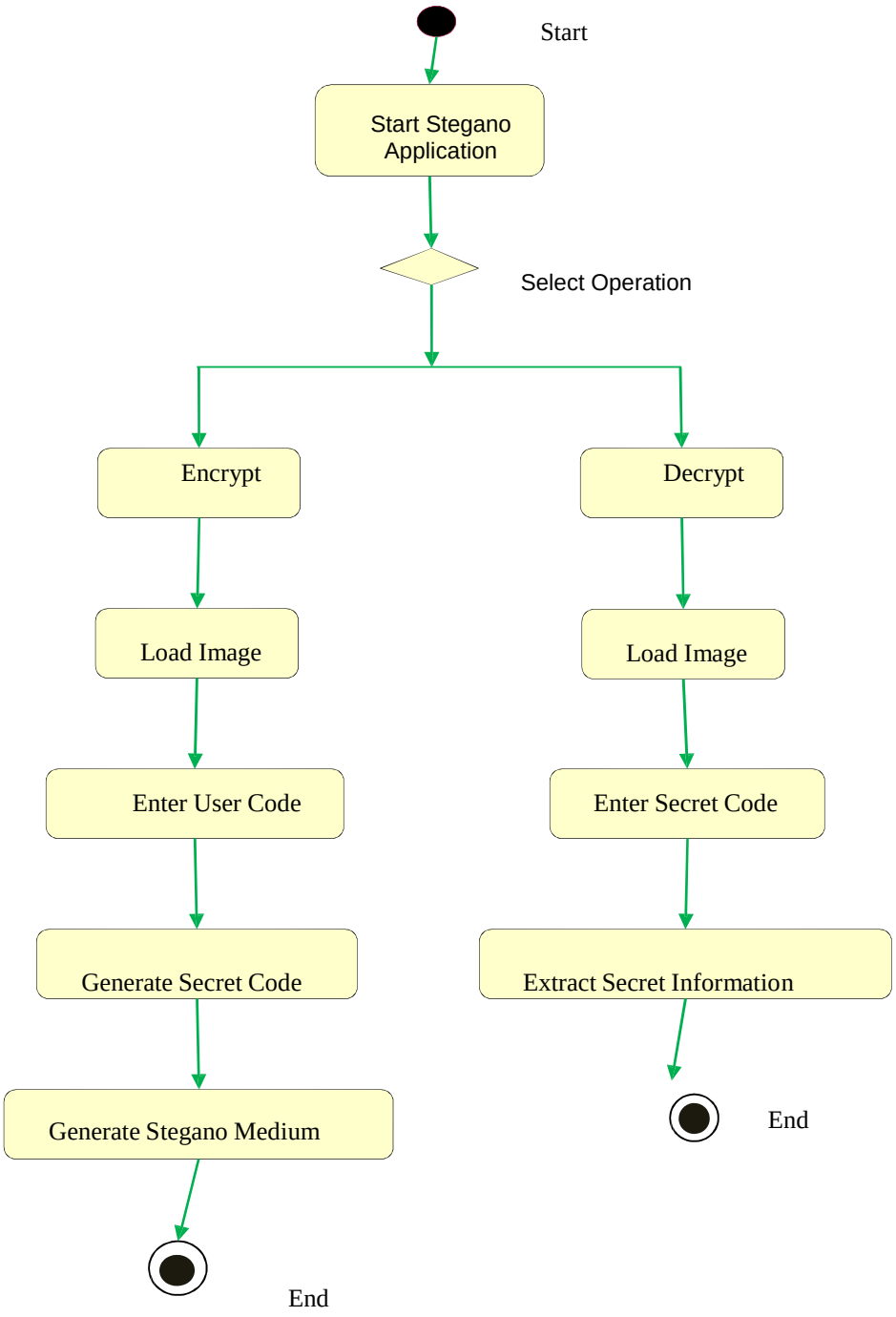


Fig: (vi)

METHODOLOGY

In this project, the methodology of encrypting the text file in an image file in order to test the accuracy and efficiency of encryption. This process helps to send the information to the authorized party without any potential risk.

The proposed method will help to secure the content within the image and encryption of audio file within the image will help to make the document much securer because even though if the unauthorized person succeeds in being able to hack the image, the person will not able to read the message as well as acquire the information in the file. In this research, we compare three steganography algorithms in order to compare the hiding capacity and efficiency of hiding the message within an image. Whenever the data is encrypted using steganography algorithms within image, neither the data nor the image embedded in should lose its originality. Hence we compare the different algorithms used for steganography for the various hiding techniques and formats and analyze the results obtained.

The process is as follows:

- Providing security for the data to be transmitted through network using steganography.
- Using digital watermarking techniques
- Implementing different stenographic algorithms
- Comparing different stenographic algorithms in means of speed, accuracy and quality of hiding.
- Proposing an approach for hiding the data within an image using a stenographic algorithm which provides better accuracy and quality of Hiding.

The IDE software is used to extensively analyze the functions of the LSB algorithm in steganography. In steganography Texts and other file formats are encrypted and embedded into an image file which is then transferred to the destination. The file's changes in resolution due to the pixels lost are analyzed for suggesting the optimal method for the technique.

RESULTS

Program User Interface

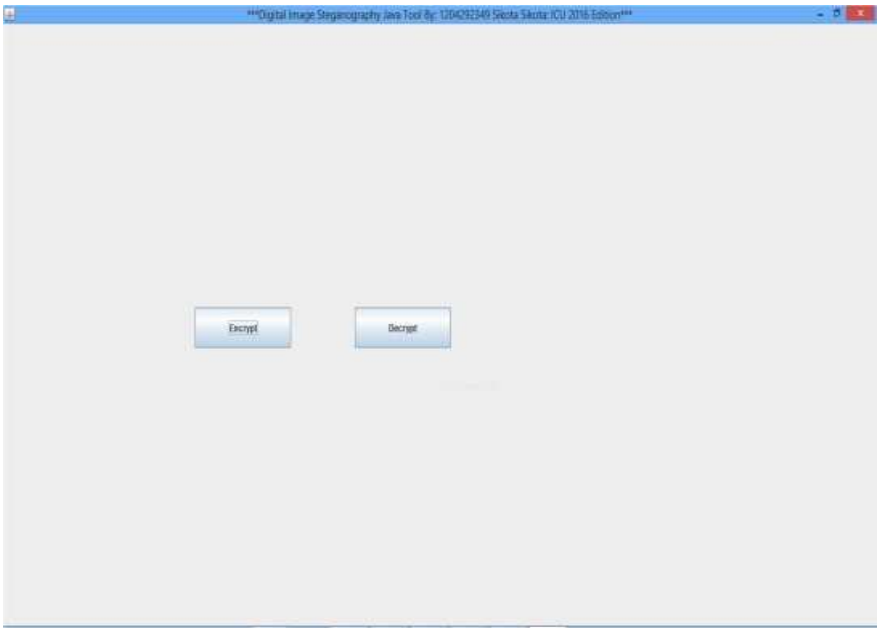


Fig: 1 shows the interface. Start process

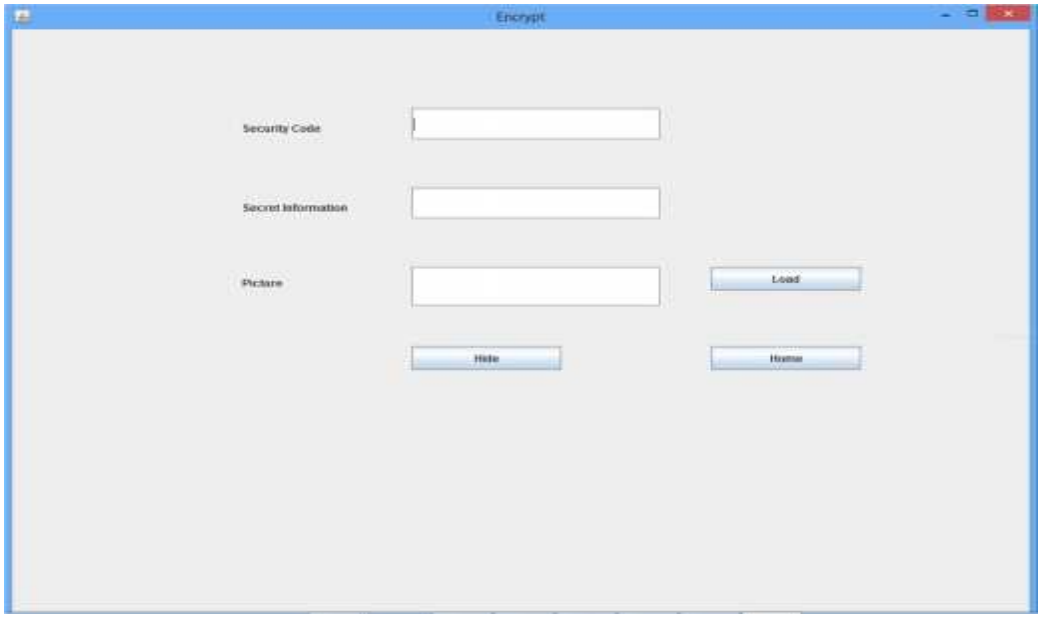


Fig: 2 enter the secret information into the dialog box

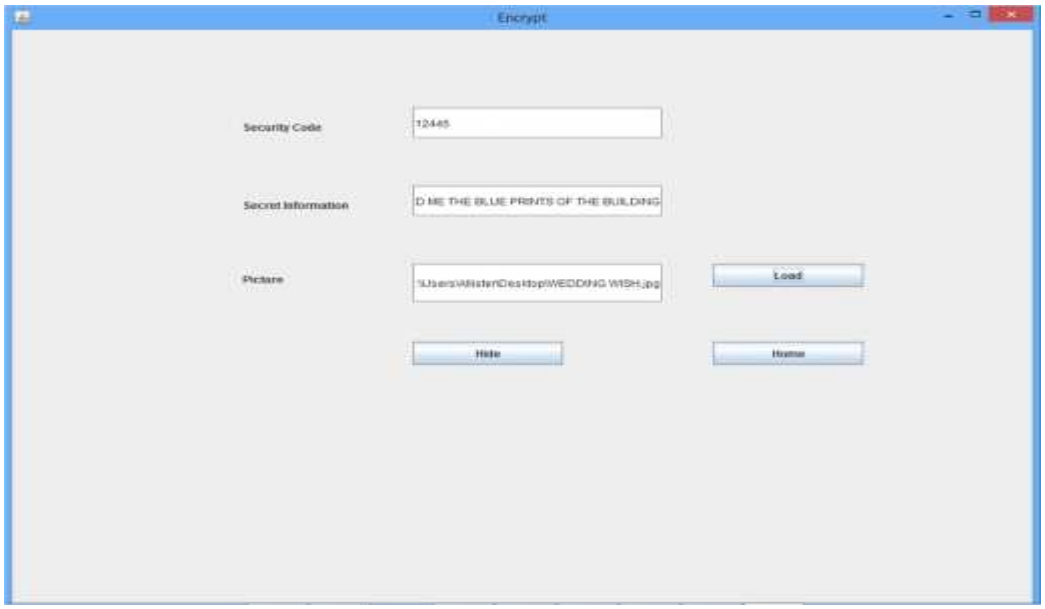


Fig: 3 enter the user code into the security code dialog box

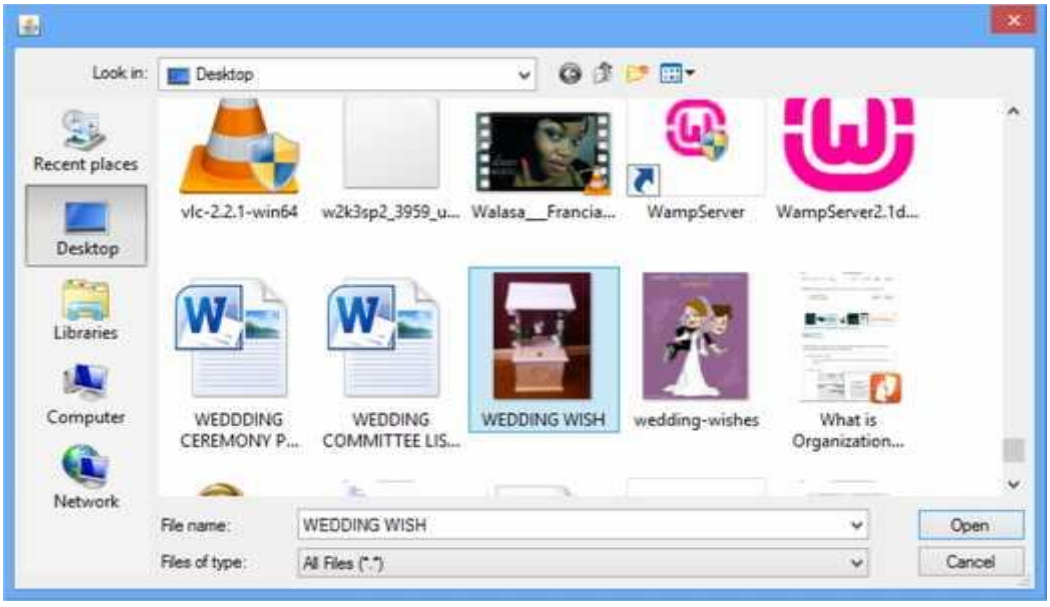


Fig: 4 Load a multimedia data, here (digital image)

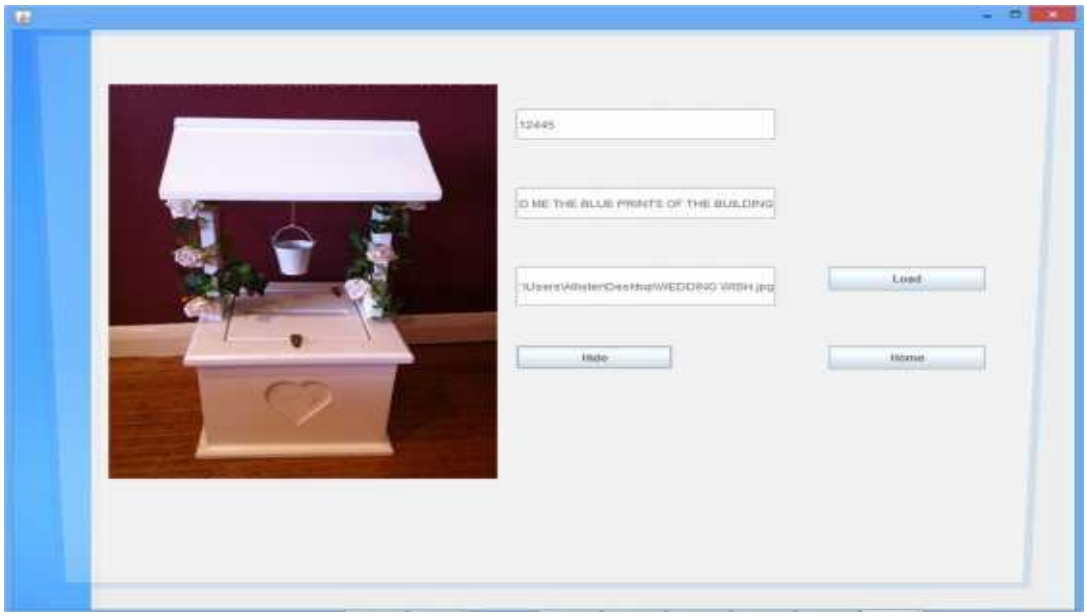


Fig: 5 Creation of secret code by user + secret information.

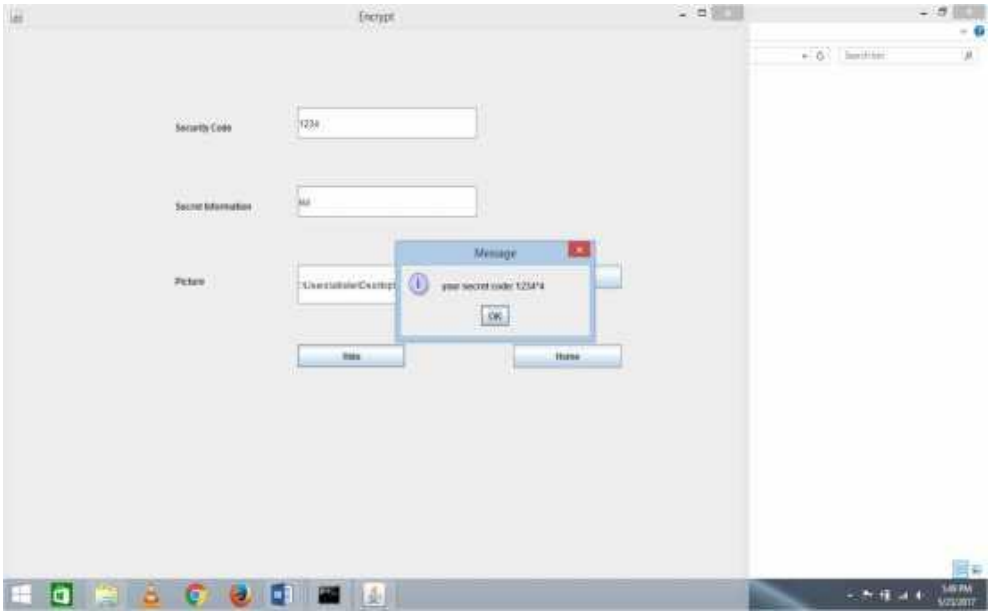


Fig: 6 hiding secret information with its security into the multimedia data

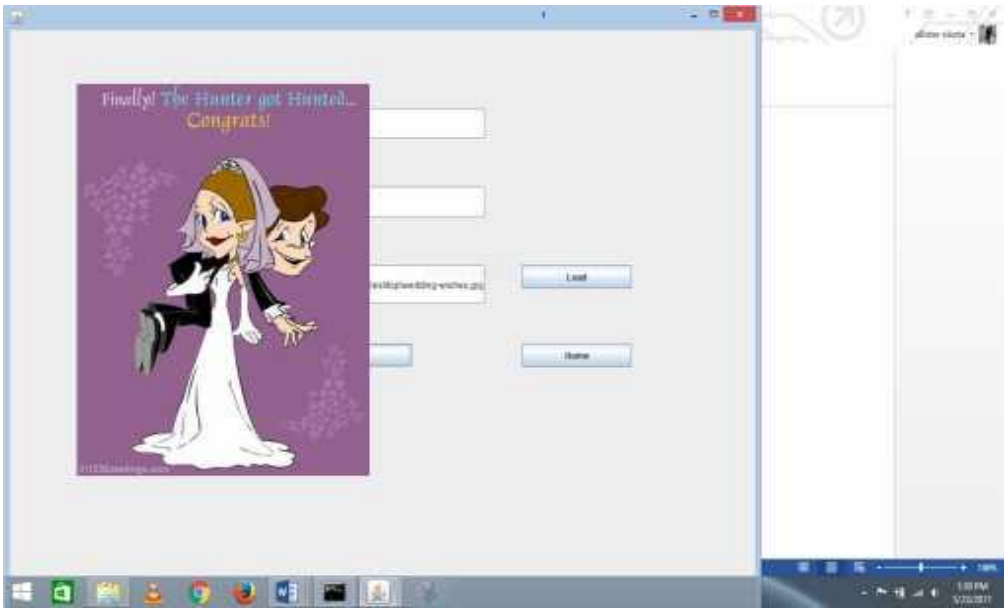


Fig: 7 A message box showing the secret key will appear. Fig: 8 close.

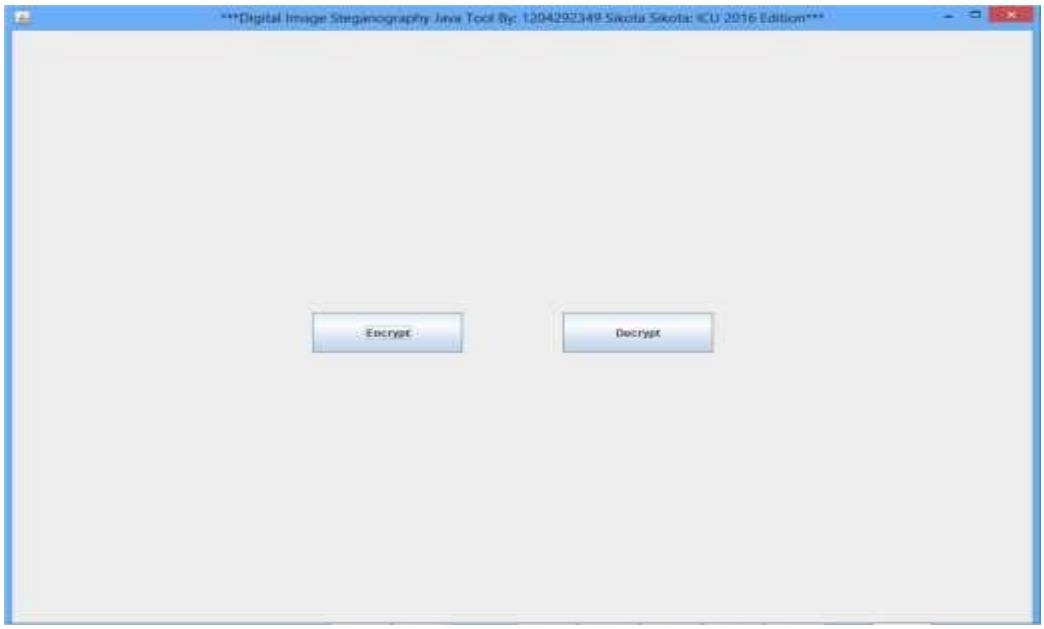


Fig: 9 start the process

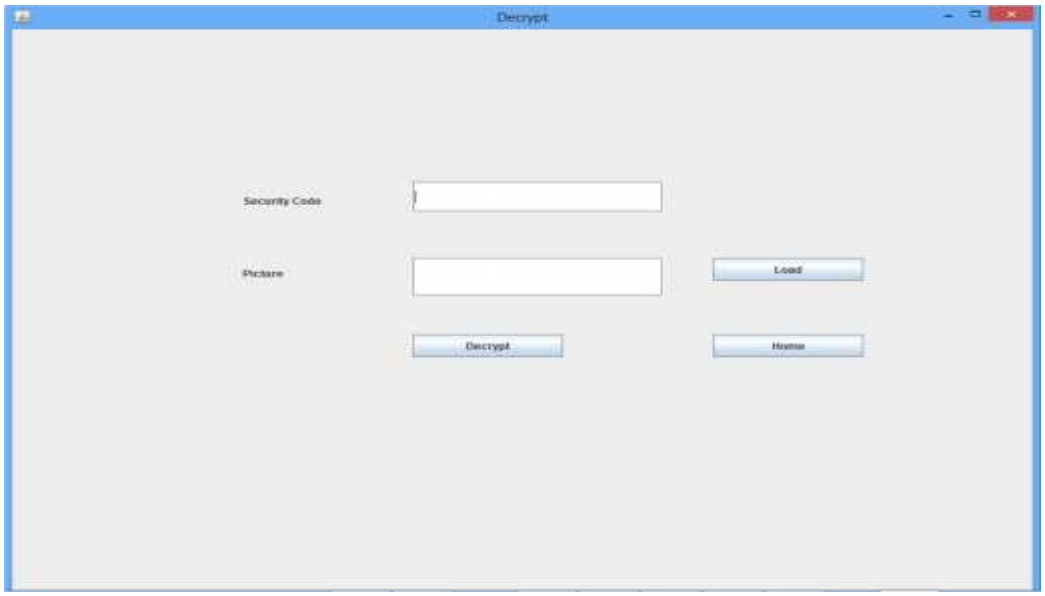


Fig: 10 enter the secret code

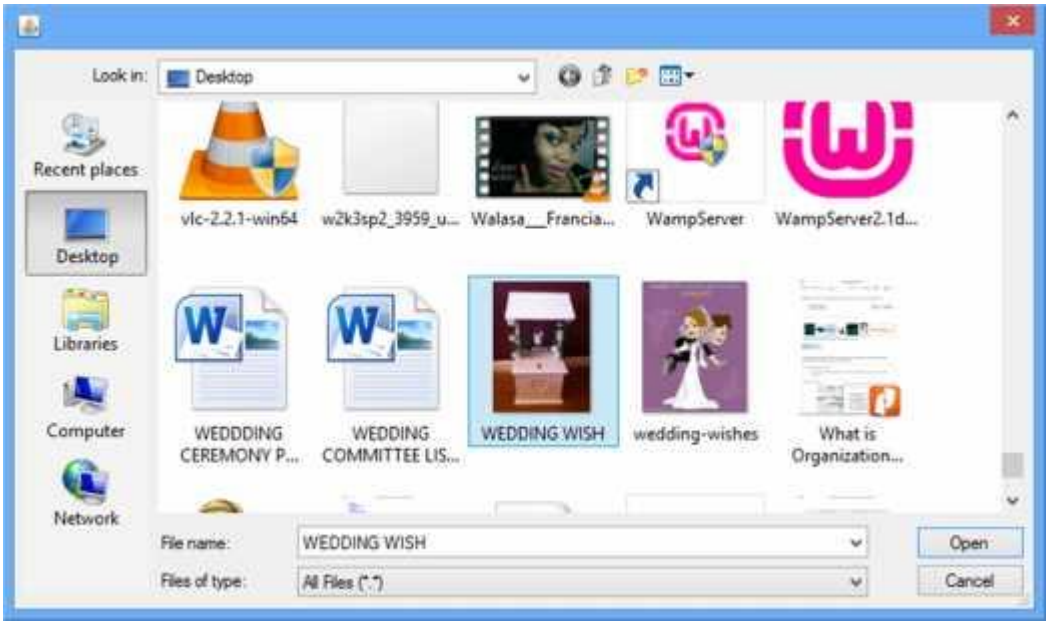


Fig: 11 Load the stego image

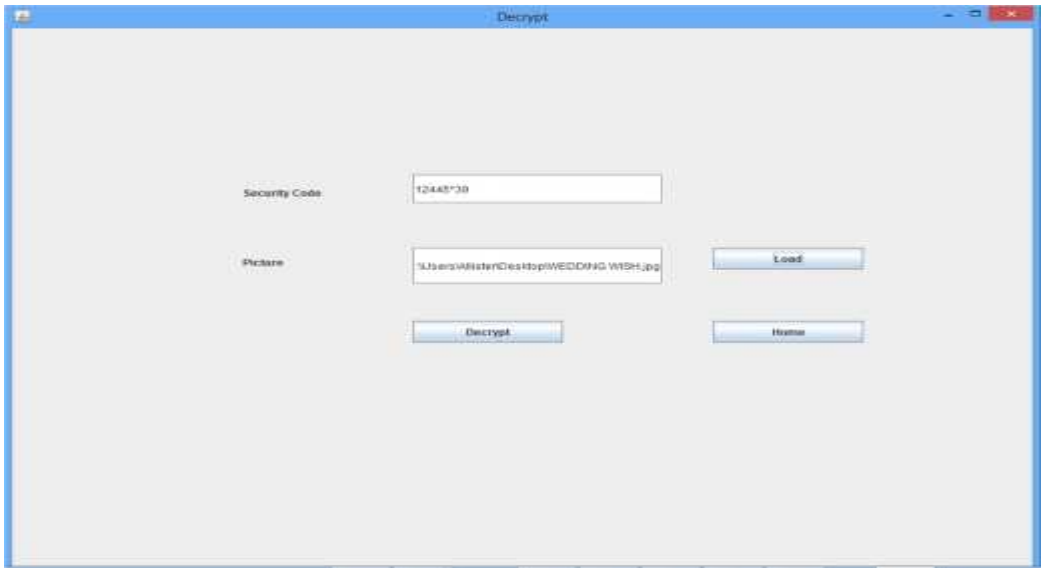


Fig: 12 extract secret information from stegano medium by using secret code

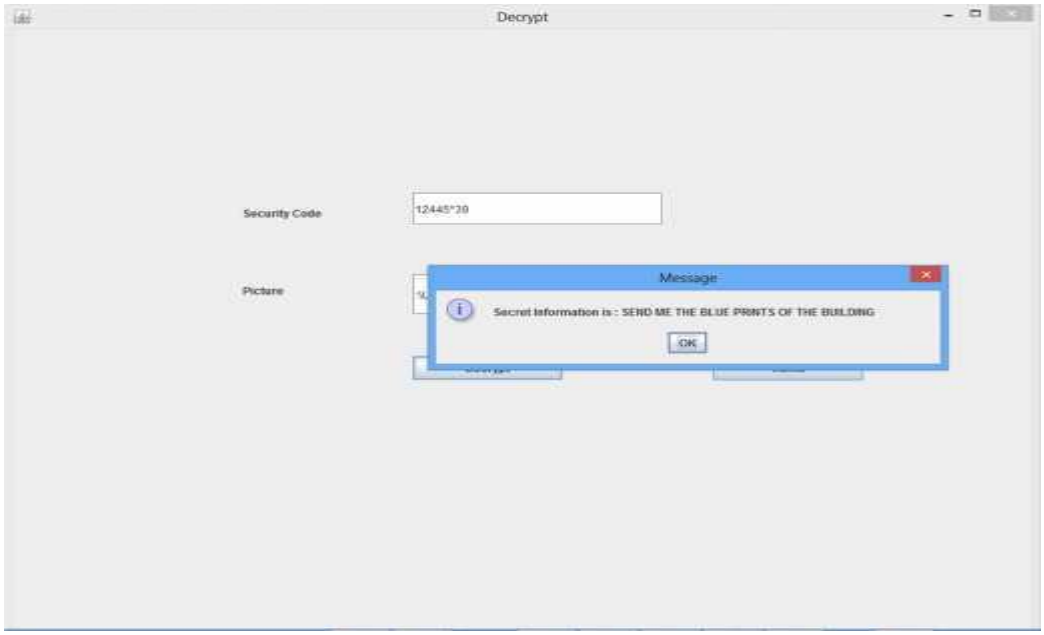


Fig: 13 secret information has been decrypted successful and it is shown in the dialog box.

Original image

stego

DISCUSSION

Based on the proposed algorithm, we develop a simple system, which implements the algorithm. We name the system as Digital Image Steganography Java Tool (DISJT). Based on the framework for the system as seen in Fig: (v). DISJT imposed on 2 layers of security. The first layer is for the security key purpose and the second layer is for the hiding and retrieving purposes. The system is introduced in. The flow chart diagram shows the main interface for the system. DISJT has two main boxes, one box for the Encryption and another box for the Decryption of data that the user hides inside the image. The image box is used for getting the image from any location, security code box is for input of the security code and it generates automatically the security code and the text box is used for input of the secret information. In order to hide the data inside the image, a secret key is required for the purpose of security reason. Fig. Shows the interface for the secret key which needs to be in 4 or 6 characters.

From Fig. 5, the secret key is required to enter once. 4 or 6 characters are used for the secret key. This secret key is also embedded inside the image together with the data. And then after encrypting the data the system generates the new code. Therefore, to reduce the size of storing the secret key inside the image, only 4 - 6 characters are used for the secret key. Once the data has been key in and the secret key has been entered and generated by the system, the new stego image can be saved to a different image file. This new stego image and the secret key generated can then be used to decrypt the secret information. Save the stego and write the generated code separately because they will be need for decryption of the secret information.

CONCLUSION

A tool implementing the steganography technique in JPEG image has been developed through java. There are two interfaces in the system: one for the encrypting (embedding) phase and the other for the decrypting (extraction). Embedding phase has several variations on the scale of security and complexity. Least Significant Bit Technique with mapping method provides more secure system since we hide an abstract intermediate New Text File in Jpeg image but this two-level mechanism also increases the complexity and our major focus was on security not complexity of the system.

The limitation of this Tool is that LSB technique requires large cover image to make usable amount of space for hiding data which are not often used on internet now days. Suspicion might rise with such images. The project was set out and come up with an application in Java that encrypts and decrypts the information.

This paper proposed a new steganography algorithm with 2 layers of security. A system named DISJT (Digital Image Steganography Java Tool) has been developed using the proposed algorithm. We tested few images with various sizes of data to be hidden. With the proposed algorithm, we found that the stego image does not have a noticeable distortion on it (as seen by the naked eyes). We also tested our stego images using PSNR value. Based on the PSNR value of each images, the stego image has a higher PSNR value. Hence this new steganography algorithm is very efficient to hide the data inside the image. DISJT can be used by various users who want to hide the data inside the image without revealing the data to other parties. DISJT maintains privacy, confidentiality and accuracy of the dat

ACKNOWLEDGEMENT

I am extremely satisfied in successfully completing the dissertation for my BSc. Information Security and Computer Forensics. I take this opportunity to thank all my faculties and mentors who took a huge part in my progress. I would especially like to thank Dr. Silumbe Richard who helped in completing the dissertation with valuable suggestions and feedback ensuring my direction is correct in my first research project. The Aims library and the Journal Access Systems were extremely helpful in providing me with the necessary knowledge to actively engage in the project.

I would like to thank my friends for helping me with their expertise in Java technologies for building the Digital image steganography application Tool. Above all, I am grateful to my Family for the support they gave me to pursue this course.

REFERENCING

[1] Alfred J, Met al., 1996. Hand book of applied Cryptography. First edn.

[2] Ali-al, H. Mohammad, A. 2010. Digital Audio Watermarking Based on the Discrete Wavelets Transform and Singular Value Decomposition, European Journal of Scientific Re- search, vol 39(1), pp 231-239.

[3] Amirthanjan, R, Aklla, R& Deeplkachowdavarapu, P., 2010, A Comparative Analysis of Image Steganography, international Journal of Computer Application, 2(3), pp.2-10.

[4] Arnold, M. 2000. Audio watermarking: Features, applications and algorithms, proceeding of the IEEE international Conference on Multimeadia and Expo, pp 1013-1016. Bandyopadhya, S.k., 2010. An Alternative Approach of Steganography Using Reference Image. International journal of Advancements in Technology, 1(1), pp.05 -11.

[5] Bloom,J. A. et all,2008, Digital watermarking and Steganography. 2nd ed. Morgan

[6] Bishop, M., 2005. Introduction to computer security. 1st ed. Pearson publications. Cachin, C., 2004. Information: Theoretic model for steganography. Work shop on information hiding, USA.

[7] Chan, C.K. Cheng, L.M., 2004. Hiding data in images by simple lsb substitution: Steganography. 2nd Ed. Elsevier.

[8] Cox, I. Miller, M. Bloom, J. Fridrich, J & Kalker, T. 2008. Digital watermarking and Steganogra- phy. 2nd Ed. Elsevier.

[9] Cummins, J. Diskin, P. Lau, S. & Parett, R., 2004. Steganography and digital watermarking. School of computer science. Vol 1.

[10] David, W. (2004) Managing information: IT for Business purpose. 3rd edn, pg no. 215, Elsevier. El-Emam,N.N., Embedding a large amount of information using high secure neural based ste- ganography algorithm. International Journal of Signal Processing 4(2), pp,5-4. www.youtube.com-Steganography
www.youtube.com-howtohidetextinanimage
http://introcs.cs.princeton.edu/java/10elements/

[11] Grover, D., 2001. Data Watermarking: Steganography and Watermarking Of Digital Data. Com- puter Law & Security Report, 17(2), pp.65 - 67.

[12] Glenford et al., 2004. The art of software testing. 2nd edn, pg no. 183, john wiley.

[13] Hellman, M.E., 2002. An overview of public key cryptography. IEEE communication maga- zine.

[14] Jeffrey A, Bloom et al., 2008. Digital watermarking and steganography, 2nd edn, Morgan Kauf- mann publications.

[15] Johnson, N.F, Jajodia, S., 1998, Exploring Steganography: seeing the unseen computing practic- es. IEEE journal, Vol 1.
[16] Kahate, A., 2008. Cryptography and network security. 2nd ed. McGraw - hill. Krenn, J.R., 2004. Steganography and Steganalysis. IEEE communication magazine.
[17] Morkel et all., 2005, An overview of image steganography, information security south Africa conference (ISSA) research group, vol. 1, no. 1.
[18] Nageswara rao, p., 2008, Software Testing concerpts and tools. 2nd edn, Dreamtec Press. Provos, N. & Honeyman, p.,2003. Hide and seek: an introduction to steganography. IEEE com- puter society.
[19] Siridevi, r. Damodaram, A. & Narasingham, S., 2009. Efficient Method of Audio Steganography by Modified Lsb Algorithm and Strong Encryption Key with Enhanced Security. Journal of theo- retical and applied information technology, 5(2), pp.25-31
[20] Stallings, W., 2007. Computer security. 5th ed. USA: Pearson education.
[21] Talele, K.T. Gandhe, S.T & Keskar, A.A., 2010. Steganography Security Fircopyright Whitman, M.E. & Mattord, H.J., 2007. Principles of information security. Thomson course tech- nology.
[22] Westfied, A. & Pfitzmann, A., 2001. Attacks on steganographic systems: breaking the ste- ganographic utilities ezstego, jsteg, steganos and s-tools. Dresden University of technology. Wikipedia., 2010. Software testing. [e-book] http//:www.wikipedia.org/softwaretesting[Retried 30 Aug2010]
[23] Yang et al., 2010. A semi – fragile watermarking algorithm using adaptive least significant bit substitution. Information technology journal, vol 9 (01), pp 20 – 26.
[24] Zaidoon Kh, A. Zaidan, A.A. Zaidan,B.B & Alanaza.h.o., 2010. Overview: main fundamentals for steganography. Journal of Computing, 2(3), pp.40-43.
[25] M. Chen, N. Memon, E.K. Wong, Data hiding in document images, in: H. Nemati (Ed.). Premier Reference Source–Information Security and Ethics: Concepts, Methodologies,
[26] Tools and Applications, New York: Information Science Reference, 2008, pp. 438-450.
[27] D.C. Lou, J.L. Liu, H.K. Tso, Evolution of information – hiding technology, in H. Nemati (Ed.), Premier Reference Source–Information Security and Ethics: Concepts, Methodologies, Tools and Applications, New York: Information Science Reference, 2008, pp. 438-450.
[28] Schneider, Secrets & Lies, Indiana:Wiley Publishing, 2000.

[29] E. Cole, Hiding in Plain Sight: Steganography and the Art of Covert Communication, Indianapolis: Wiley Publishing,

[30] T. Jahnke, J. Seitz, (2008). An introduction in digital watermarking applications, principles and problems, in: H. Nemati (Ed), Premier Reference Source–Information Security and Ethics: Concepts, Methodologies, Tools and Applications, New York: Information Science Reference, 2008, pp. 554-569.

[31] M. Warkentin, M.B. Schmidt, E. Bekkering, Steganography and steganalysis, Premier reference Source–Intellectual Property Protection for Multimedia Information technology, Chapter XIX, 2008, pp. 374-380.

[32] N.N. El-Emam, Hiding a large amount of data with high security using steganography algorithm, Journal of Computer Science 3 (2007) 223-232.

[33] P.Y. Chen, W.E. Wu, A modifed side match scheme for image steganography, International Journal of Applied Science & Engineering 7 (2009) 53-60.

[34] C.C. Chang, H.W. Tseng, A steganographic method for digital image using side match, Pattern Recognition Letters 25 (2004) 1431-1437.

[35] P.C. Wu, W.H. Tsai, A steganographic method for images by pixel-value differencing, Pattern Recognition Letters 24 (2003) 1613-1626.

[36] R. Ibrahim and T.S. Kuan, Digital Image Steganography Java Tool (DISJT): hiding secret message inside an image, Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2010, San Francisco, USA, 2010, pp. 144-148.

TABLES AND FIGURES

Program Source Codes

HOME import
java.awt.*; import
javax.swing.*; import
java.awt.event.*;

public class Home extends JFrame implements ActionListener
{
/**
*
*/
private static final long serialVersionUID = 1L;
private JButton encrypt,decryptmsg;

Home()
{
super("***Digital Image Steganography Java Tool By: 1204292349 Sikota Sikota: ICU 2016 Edition***");
Container con=getContentPane();
con.setLayout(null); encrypt=new
JButton("Encrypt");
encrypt.addActionListener(this);
encrypt.setBounds(300,350,150,50);
decryptmsg=new JButton("Decrypt");
decryptmsg.addActionListener(this);
decryptmsg.setBounds(550,350,150,50);
con.add(encrypt);
con.add(decryptmsg);
}

public void actionPerformed(ActionEvent ae)
{
if(ae.getSource()==encrypt)
{
this.dispose(); EncryptPage ep=new
EncryptPage();
ep.setSize(1035,790);
ep.setVisible(true);
}

if(ae.getSource()==decryptmsg)
{
this.dispose(); DecryptPage dp=new
DecryptPage(); dp.setSize(1035,790);
dp.setVisible(true);
}
}

public static void main(String args[])

```
{
Home h=new Home();
h.setSize(1035,790);
h.setVisible(true);
}
}
```

ENCRYPT PAGE

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.io.*;
import java.awt.image.*;
import com.sun.image.codec.jpeg.*;

public class EncryptPage extends JFrame implements ActionListener
{
/**
*
*/
private static final long serialVersionUID = 1L;
private JLabel code_label,secret_label,picture_label;
private JTextField code_text,secret_text,picture_text;
private JButton picture_load_button,hide_button,home_button;
String filepath="",secret_code="",secret_info="",user_key="";
Container con=null;
JLabel jl;
byte img_byte[]=new byte[6000];
FileDialog fd;

//////// Variables for creating an image from an integer array  ////////////////////////

Image img;
Dimension d;
int iw,ih;
int w=10,h=10;
int pix[];
int hist[]=new int[256];
int t[];
int max_hist=0;
boolean ok;
static Image newimg;
int key,k;

EncryptPage()
{ super("Encrypt");
con=getContentPane();
con.setLayout(null);

code_label=new JLabel("Security Code");
```

```
code_label.setBounds(230,100,150,50);
code_text=new JTextField(200);
code_text.setBounds(400,100,250,40);
secret_label=new JLabel("Secret Information");
secret_label.setBounds(230,200,150,50);
secret_text=new JTextField(200);
secret_text.setBounds(400,200,250,40);

picture_label=new JLabel("Picture");
picture_label.setBounds(230,300,250,40);
picture_text=new JTextField(200);
picture_text.setBounds(400,300,250,50);
picture_load_button=new JButton("Load");
picture_load_button.setBounds(700,300,150,30);
picture_load_button.addActionListener(this);

hide_button=new JButton("Hide");
hide_button.setBounds(400,400,150,30);
hide_button.addActionListener(this);
home_button=new JButton("Home");
home_button.setBounds(700,400,150,30);
home_button.addActionListener(this);

jl=new JLabel();
jl.setBounds(700,500,150,30);

fd=new FileDialog(new JFrame());

con.add(code_label);
con.add(code_text);
con.add(secret_label);
con.add(secret_text);
con.add(picture_label);
con.add(picture_text);
con.add(picture_load_button);
con.add(hide_button);
con.add(home_button);
//con.add(jl);
}

public void actionPerformed(ActionEvent ae)
{
if(ae.getSource()==picture_load_button)
{ fd.setVisible(true);
filepath=fd.getDirectory()+fd.getFile();
picture_text.setText(filepath);
}else if(ae.getSource()==hide_button)
{
int starflag=0;
secret_code=code_text.getText();
for(int i=0;i<secret_code.length();i++)
```

```
{
if(secret_code.charAt(i)=='*')
{
starflag=1;
}
}
if(starflag==0)
{ secret_info=secret_text.getText();
user_key=secret_code+"*"+new String(""+secret_info.length());
System.out.println("user key  :"+user_key);
String secret_code_info=user_key+"*"+secret_info+"*";
byte secret_byte_array[]=secret_code_info.getBytes();
int secret_int_array[]=new int[secret_byte_array.length];

try{
if(filepath.equals("") && (secret_text.getText()).equals(""))
JOptionPane.showMessageDialog(null,"image and secret info are empty. enter them");
else if(secret_info.length()==0 && filepath.length()>0)
JOptionPane.showMessageDialog(null,"enter secret info");
else if(filepath.length()==0 && (secret_text.getText()).length()>0)
JOptionPane.showMessageDialog(null,"load an image");
else
{
ImageIcon ic=new ImageIcon(filepath);
img=ic.getImage();
iw=img.getWidth(null);
ih=img.getHeight(null); pix=new
int[iw*ih];
t=new int[iw*ih];
PixelGrabber pg=new PixelGrabber(img,0,0,iw,ih,pix,0,iw);
ColorModel cm=pg.getColorModel();
int ww=pg.getWidth();
int hh=pg.getHeight();
pg.grabPixels();

key=secret_byte_array.length;
int k=key;
int j=0;

for(int i=0;i<pix.length;i++)
{
if((i%20)==0 && k>0)
{ secret_int_array[j]=(int)secret_byte_array[j];
System.out.println("user key :"+secret_int_array[j]);
pix[i]=secret_int_array[j];
j++;
k--;
}
}
newimg =con.createImage(new MemoryImageSource(ww,hh,cm,pix, 0, ww));
```

```
jl.setIcon(new ImageIcon(newimg));
JOptionPane.showMessageDialog(null,"your secret code: "+user_key+"");

MediaTracker mediaTracker = new MediaTracker(new Container());
mediaTracker.addImage(newimg, 0); mediaTracker.waitForID(0);

int thumbWidth = 400;//Integer.parseInt(400);
int thumbHeight = 400;//Integer.parseInt(400);
double thumbRatio = (double)thumbWidth / (double)thumbHeight;
int imageWidth = newimg.getWidth(null);
int imageHeight = newimg.getHeight(null);
double imageRatio = (double)imageWidth / (double)imageHeight;

if (thumbRatio < imageRatio)
{
thumbHeight = (int)(thumbWidth / imageRatio);
}
else
{
thumbWidth = (int)(thumbHeight * imageRatio);
}

// draw original image to thumbnail image object and
// scale it to the new size on-the-fly
BufferedImage thumbImage = new BufferedImage(newimg.getWidth(null), newimg.getHeight(null),
BufferedImage.TYPE_INT_RGB); Graphics2D graphics2D =
thumbImage.createGraphics();
graphics2D.setRenderingHint(RenderingHints.KEY_INTERPOLATION,
RenderingHints.VALUE_INTERPOLATION_BILINEAR);
graphics2D.drawImage(newimg, 0, 0, newimg.getWidth(null), newimg.getHeight(null), null);
// save thumbnail image to OUTFILE
File f=new File("secpic.jpg"); BufferedOutputStream out =
new BufferedOutputStream(new FileOutputStream(f));
JPEGImageEncoder encoder = JPEGCodec.createJPEGEncoder(out);
JPEGEncodeParam param = encoder.
getDefaultJPEGEncodeParam(thumbImage);
int quality = 80;//Integer.parseInt(args[4]);
quality = Math.max(0, Math.min(quality, 100));
param.setQuality((float)quality / 100.0f, false);
encoder.setJPEGEncodeParam(param);
encoder.encode(thumbImage);
out.close();
System.out.println("Done.");

test t=new test(newimg);
t.setSize(1035,790);
t.setVisible(true);
}
}catch(Exception e)
{
```

```
System.out.println(e);
}
}else
JOptionPane.showMessageDialog(null,"Do not enter '*' in secrect code");
}else
{
this.dispose(); Home
h=new Home();
h.setSize(1035,790);
h.setVisible(true);
}
}

public static void main(String args[])
{
EncryptPage ep=new EncryptPage();
ep.setSize(1035,740);
ep.setVisible(true);
}
}
```

DECRYPT PAGE

```
import java.awt.Container;
import java.awt.Dimension;
import java.awt.FileDialog;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.image.ColorModel;
import java.awt.image.MemoryImageSource;
import java.awt.image.PixelGrabber;
//import javax.imageio.stream.*;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextField;

public class DecryptPage extends JFrame implements ActionListener
{
/**
*
*/
private static final long serialVersionUID = 1L;
private JLabel code_label,picture_label; private
JTextField code_text,picture_text;
private JButton picture_load_button,decrypt_button,home_button;
String filepath="",secret_code="",secret_info=""; Container
```

```
con=null;
JLabel jl;
byte img_byte[]=new byte[6000];
FileDialog fd;


//////// Variables for creating an image from an integer array  ////////////////////////

Image img;
Dimension d;
int iw,ih;
int w=10,h=10;
int pix[];
int hist[]=new int[256];
int t[];
int max_hist=0;
boolean ok;
Image newimg;
int key,k;
String user_key="";

DecryptPage()
{ super("Decrypt");
con=getContentPane();
con.setLayout(null);

code_label=new JLabel("Security Code");
code_label.setBounds(230,200,150,50);
code_text=new JTextField(200);
code_text.setBounds(400,200,250,40);

picture_label=new JLabel("Picture");
picture_label.setBounds(230,300,250,40);
picture_text=new JTextField(200);
picture_text.setBounds(400,300,250,50);
picture_load_button=new JButton("Load");
picture_load_button.setBounds(700,300,150,30);
picture_load_button.addActionListener(this);

decrypt_button=new JButton("Decrypt");
decrypt_button.setBounds(400,400,150,30);
decrypt_button.addActionListener(this);
home_button=new JButton("Home");
home_button.setBounds(700,400,150,30);
home_button.addActionListener(this);

jl=new JLabel();
jl.setBounds(700,500,150,30);

fd=new FileDialog(new JFrame());
```

```
con.add(code_label);
con.add(code_text);
con.add(picture_label);
con.add(picture_text);
con.add(picture_load_button);
con.add(decrypt_button);
con.add(home_button);
con.add(jl);
}

public void actionPerformed(ActionEvent ae)
{
if(ae.getSource()==picture_load_button)
{ fd.setVisible(true);
filepath=fd.getDirectory()+fd.getFile();
picture_text.setText(filepath);
}else if(ae.getSource()==decrypt_button)
{
String sc=code_text.getText();
int star_flag=0;
String star_value="";
for(int i=0;i<sc.length();i++)
{
if(sc.charAt(i)=='*')
star_flag=1;
if(star_flag==1&& star_flag!=2)
{
i= ++i;
star_value=sc.substring(i);
star_flag=2;
}
}
System.out.println("star value er:"+Integer.parseInt(star_value));
k=sc.length()+1+Integer.parseInt(star_value);
try{
img=EncryptPage.newimg;
key=k;
System.out.println("key ckeck in temp:"+key);
user_key=sc;

Container con=getContentPane();

iw=img.getWidth(null);
ih=img.getHeight(null);
pix=new int[iw*ih];
t=new int[iw*ih];

PixelGrabber pg=new PixelGrabber(img,0,0,iw,ih,pix,0,iw);
ColorModel cm=pg.getColorModel();
int ww=pg.getWidth();
```

```
int hh=pg.getHeight();
pg.grabPixels();

int secret_check[]=new int[sc.length()];
byte sc_byte[]=sc.getBytes();

for(int i=0;i<sc.length();i++)
secret_check[i]=sc_byte[i];

int secret_info[]=new int[key];
byte b[]=new byte[key];
int j=0,loop=0,flag=0,star2_flag=0;

System.out.println("hi welcome");

for(int i=0;i<pix.length;i++)
{
if((i%20)==0 && k>0 && flag==0)
{

System.out.println("one");

if(loop<user_key.length() && secret_check[loop]==pix[i] && star2_flag<2)
{ System.out.println("two");
if((char)secret_check[loop]=='*')
{
star2_flag++;
}

k--;
loop++;
}else if(star2_flag>=1)
{
System.out.println("else if");
secret_info[j]=pix[i];
b[j]=(byte)pix[i];
System.out.println("secrect pix :"+new String(""+(char)b[j])+"");
j++;
k--;
}
else
{ System.out.println("star flag
:"+star2_flag); System.out.println("else");
flag=1;
}
}
}
if(flag==0)
{
String s=new String(b);
```

```
s=new String(s.substring(1));

System.out.println("secret information :"+s);
System.out.println("key  :"+key);
JOptionPane.showMessageDialog(null,"Secret Information is : "+s);
}
else JOptionPane.showMessageDialog(null,"code you entered is not valid");
newimg =con.createImage(new MemoryImageSource(ww,hh,cm,pix, 0, ww));
}catch(Exception e)
{
System.out.println(e);
}
}else
{
this.dispose(); Home
h=new Home();
h.setSize(1035,790);
h.setVisible(true);
}
}
public static void main(String args[])
{
DecryptPage dp=new DecryptPage();
dp.setSize(1035,740);
dp.setVisible(true);
}
}
```

TEST

```
import java.awt.Graphics;
import java.awt.Image;

import javax.swing.JFrame;

public class test extends JFrame
{
/**
*
*/
private static final long serialVersionUID = 1L;
Image newimg;
test()
{
}
test(Image m)
{
newimg=m;
}
public void paint(Graphics g)
{
```

```
g.drawImage(newimg,100,100,null);
}

public static void main(String args[])
{
test bp=new test();
bp.setSize(1035,740);
bp.setVisible(true);
}
}
```