IMPLEMENT SECURE DATA TRANSFER USING STEGANOGRAPHY

A Project submitted to the University of Madras in partial fulfillment of requirements for the award of the degree Master of Computer Applications

Submitted by

NIRMALA M

Reg. No. 811800033

Under the Guidance of

Dr. A. JAYASUDHA M.Sc., M.Phil., Ph.D.,Assistant Professor
Associate Professor
Department of Computer Science & Applications

SRM ARTS AND SCIENCE COLLEGE
KATTANKULATHUR – 603 203

UNIVERSITY OF MADRAS
DECEMBER 2020

BONAFIDE CERTIFICATE

This is to certify that the project report entitled

" IMPLEMENT SECURE DATA TRANSFER USING STEGANOGRAPHY "

being submitted to the University of Madras, Chennai - 600 005

by

NIRMALA M

(Register number: 811800033)

for the partial fulfillment for the award of the degree

MASTER OF COMPUTER APPLICATIONS

is the bonafide record work carried by her,

under my guidance and supervision

Signature of the Guide **Head of the Department**

Submitted for the Viva-Voce Examination held on ………………………..………..at SRM Arts and Science College, Kattankulathur – 603 203.

Date: **Examiners**

1.

2.

ACKNOWLEDGEMENT

I would like to express my thanks to **Dr. T. R. PAARIVENDHAR, B.Sc., M.I.E,** Chairman and **Dr. R. VASUDEVARAJ, M.Com., M.Phil., B.Ed., MBA., Ph.D.,** Principal of our college, for their scholarly guidance and motivation throughout the project work as well as the curriculum.

My sincere thanks to **Dr. G. PARTHIBAN, M.C.A., M.Phil., Ph.D.,** Associate Professor, Department of Computer Science & Applications for the useful guidance and helpful mind to complete this project.

My sincere thanks to my guide **Dr. A. JAYASUDHA M.Sc., M.Phil., Ph.D**., Assistant Professor, Department of Computer Science & Applications, for suggesting the problem, offering inspiring guidance and fruitful discussion throughout the course of the work.

My special thanks to all the staff members, Department of Computer Science & Applications for their support and encouragement for the successful completion of the project.

I acknowledge my profound thanks to my parents for their encouragement and for social and economical support for completing this project work.

(NIRMALA M)

ABSTRACT

Steganography is the art of hiding the fact that communication is taking place, by hiding information in other information. Many different carrier file formats can be used, but digital images are the most popular because of their frequency on the internet. For hiding secret information in images, there exists a large variety of steganography techniques some are more complex than others and all of them have respective strong and weak points. Different applications may require absolute invisibility of the secret information, while others require a large secret message to be hidden. This project report intends to give an overview of image steganography, its uses and techniques. It also attempts to identify the requirements of a good steganography algorithm and briefly reflects on which steganographic techniques are more suitable for which applications.

1. INTRODUCTION

One of the reasons that intruders can be successful is the most of the information they acquire from a system is in a form that they can read and comprehend. Intruders may reveal the information to others, modify it to misrepresent an individual or organization, or use it to launch an attack. One solution to this problem is, through the use of steganography. Steganography is a technique of hiding information in digital media. In contrast to cryptography, it is not to keep others from knowing the hidden information but it is to keep others from thinking that the information even exists.

Steganography become more important as more people join the cyberspace revolution. Steganography is the art of concealing information in ways that prevents the detection of hidden messages. Steganography include an array of secret communication methods that hide the message from being seen or discovered.

Due to advances in ICT, most of information is kept electronically. Consequently, the security of information has become a fundamental issue. Besides cryptography, steganography can be employed to secure information. In cryptography, the message or encrypted message is embedded in a digital host before passing it through the network, thus the existence of the message is unknown. Besides hiding data for confidentiality, this approach of information hiding can be extended to copyright protection for digital media: audio, video and images.

The growing possibilities of modern communications need the special means of security especially on computer network. The network security is becoming more important as the number of data being exchanged on the internet increases. Therefore, the confidentiality and data integrity requires to protect against unauthorized access and use. This has resulted in an explosive growth of the field of information hiding

Steganography hide the secrete message within the host data set and presence imperceptible and is to be reliably communicated to a receiver. The host data set is purposely corrupted, but in a covert way, designed to be invisible to an information analysis.

1.1 ABOUT THE PROJECT

This project is developed for hiding information in any image file. The scope of the project is implementation of steganography tools for hiding information includes any type of information file and image files and the path where the user wants to save Image and extruded file.

MODULE DESCRIPTION

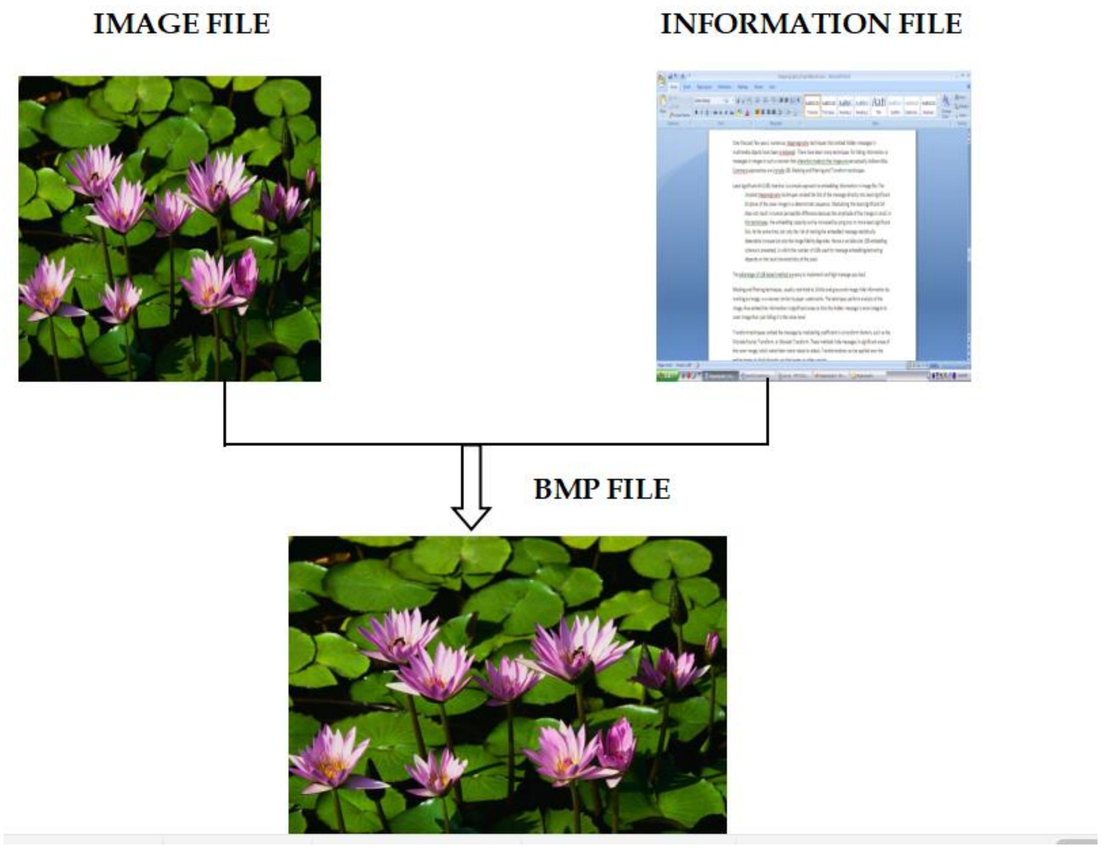
User needs to run the application. The user has two tab options – encrypt and decrypt. If user select encrypt, application give the screen to select image file, information file and option to save the image file. If user select decrypt, application gives the screen to select only image file and ask path where user want to save the secrete file.

This project has two methods – Encrypt and Decrypt.

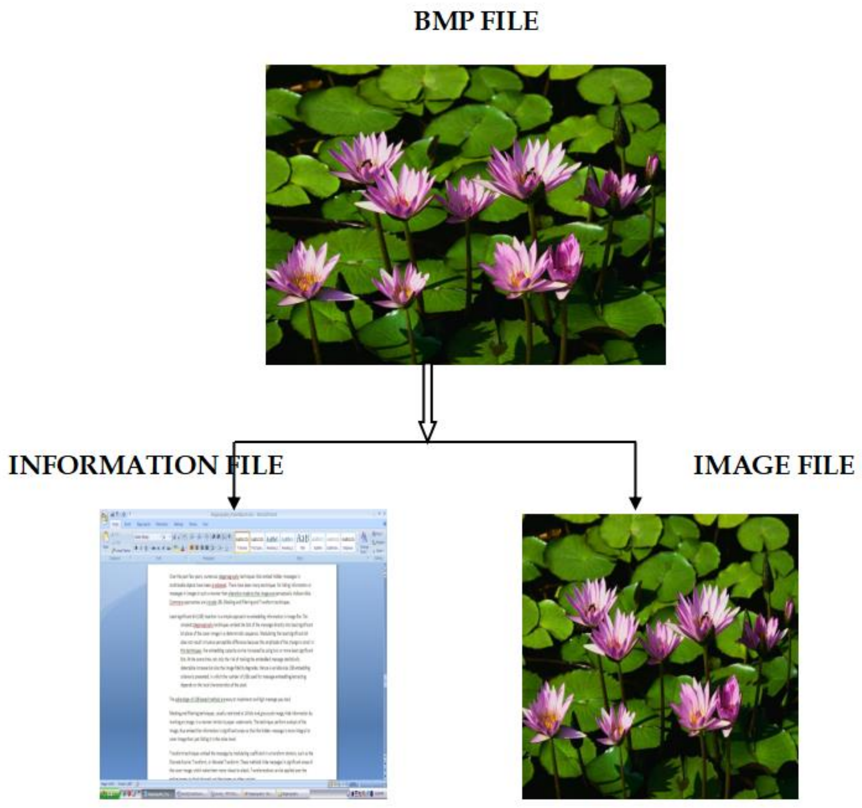
In encryption the secrete information is hiding in any type of image file.

Decryption is getting the secrete information from image file.

ENCRYPTION PROCESS



DECRYPTION PROCESS



2. SYSTEM STUDY

2.1 EXISTING SYSTEM

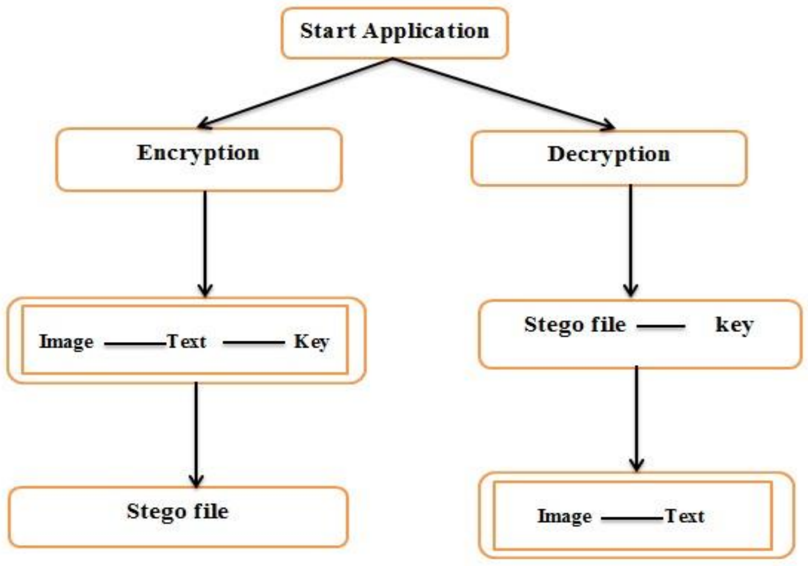
Security has always been a bedrock for transaction and protection of human lives and until the existence of artificial intelligence, humans have learned to trust themselves for each other safety or safety of information except for few times where security was threatening by breach of trust or betrayal by personal which often lead to enormous crisis. For every system and situation available there was a means of security available. For instance, indigenously, birds were sent on errands to neighbouring villages or towns to announce either the birth or death of a royal blood, spies and allies where used by kings to convey very confidential messages within and around the village, Aroko (this is a material used mostly in the western part of Nigeria especially amongst the Yoruba's) it was a symbol of information which can only be interpreted by the intended receiver despite receipt by another person, these and many more secured ways were explored in the ancient times before the advancement of modern technology.

Increasingly, as more and more situations called for stricter and tighter means of security new and inventive ways were created. A real life example is seen in the Banking sector where forms containing security questions were used in the release of funds. The banks required customers to fill forms and provide correct answers to the personal questions which apparently only the rightful owners knew, this method was used for a while till fraud became too difficult to control and banks started to invent passcodes which only rightful owners knew and gave them access to their funds. These and many more advancing security methods have been employed by banks in order to guarantee maximum security and reduce risk to its minimum.

Other means of ensuring security include sworn secrecy, oaths and many more before the advent of artificial intelligence. These measures and forms of security had several loop holes which flawed them and made security less efficient. Taking password code as an example, it was possible for an attacker to guess the random selection of numbers or letters used in forming the password thereby defeating the security and carting away confidential information, People have always wanted to keep their words and thoughts secret, hidden from prying eyes and as such, the options available to keep secrets safe are either dismal, or draw unwanted attention to the presence of a locked secret; hence, they introduced an approach for securing and safely hiding things in an unsuspecting manner. Hence, the inception of steganography The most common methods to make alterations involve the use of the Least Significant bit LSB for masking, filtering and transformations on the cover image; the art and science of making communication unintelligent to all except the intended recipient

2.2 PROPOSED SYSTEM

The proposed system deals with information and a means of securing it. It uses one of the methods of information security; Steganography. The rapid development in the transfer of data through internet made it easier to transfer data accurate and faster to the destination. Security of information is one of the important factors of information technology and communication. Steganography is art and science of invisible communication. Steganography is the method through which existence of the message can be kept secret. This is accomplished through hiding information in other information, thus hiding the existence of the communicated information.



2.3 FEASIBILITY STUDY

Technical Feasibility

The Technical issue usually raised during the feasibility stage of the investigation includes the following:

• Does the necessary technology exist to do what is suggested?

• Do the proposed equipments have the technical capacity to hold the data required to use the new system?

• Can the system be upgraded if developed? Are there technical guarantees of accuracy, reliability, ease of access and data security?

Operational Feasibility

a) **Reliability**

   The package wills pick-up current transactions on line. Regarding the old transactions.

b) **Portability**

   The application will be developed using C# .Net language etc these software will work both on Windows and Linux o/s. Hence portability problems will not arise.

c) **Availability**

   This software will be available always.

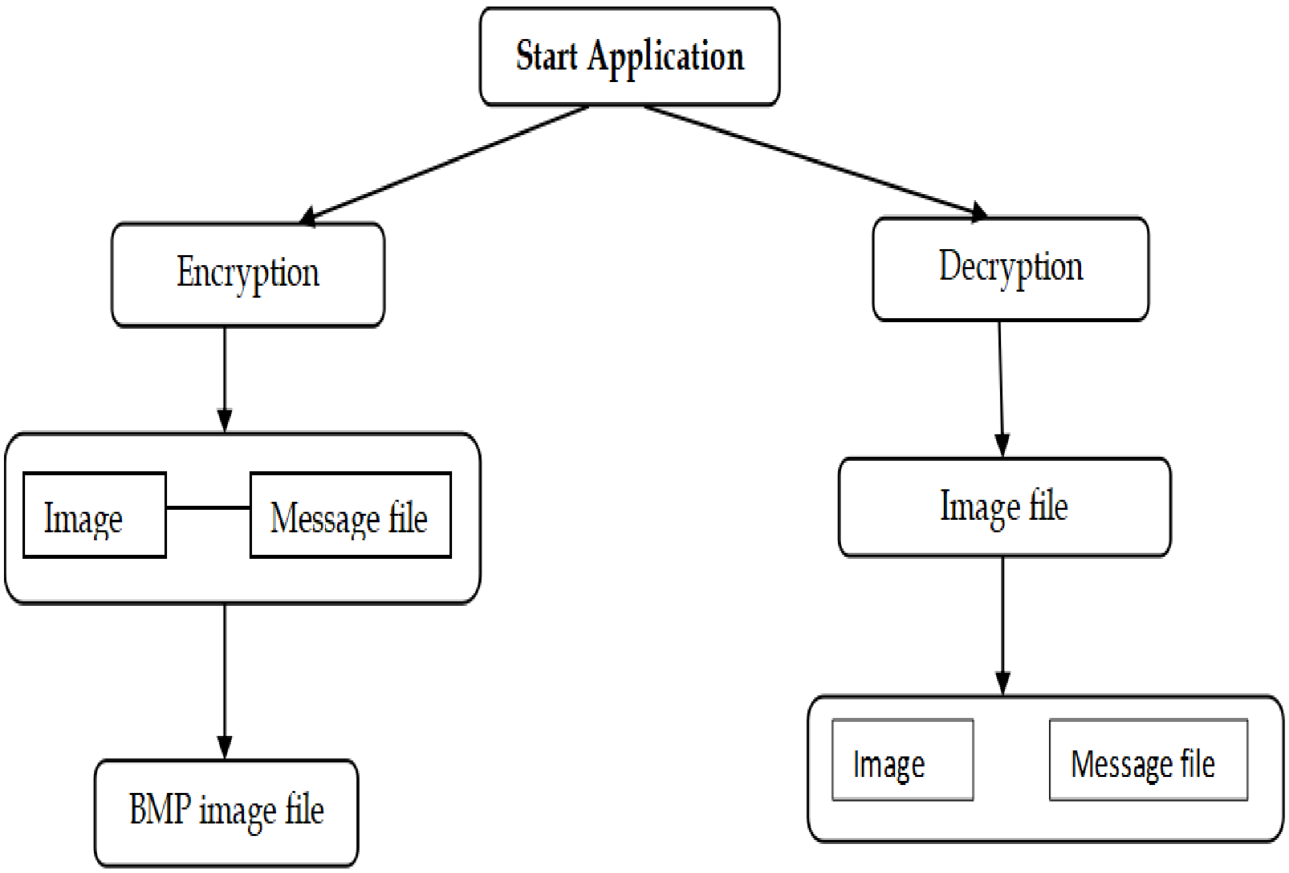
Economic Feasibility

The application takes care of the present existing system's data flow and procedures completely. It should be built as a standalone application. This is required as the encryption/decryption can be done at anywhere possible.

3. SOFTWARE PROJECT PLAN

3.1 BUSINESS DIAGRAM

The graphical representation of this system is as follows:



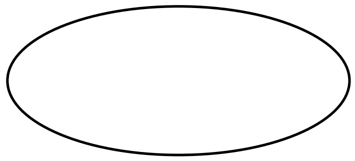
4. SYSTEM ANALYSIS

4.1 DFD DIAGRAM

A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish.

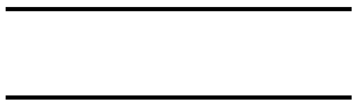
DFD symbols

1) **Process**



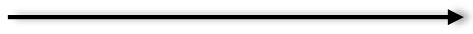
The process (function, transformation) is part of a system that transforms inputs to outputs. The symbol of a process is a circle, an oval, a rectangle or a rectangle with rounded corners (according to the type of notation). The process is named in one word, a short sentence, or a phrase that is clearly to express its essence.

2) **Warehouse**



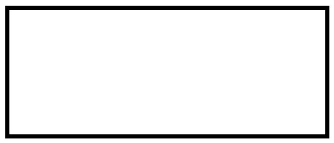
The warehouse (datastore, data store, file, database) is used to store data for later use. The symbol of the store is two horizontal lines, the other way of view is shown in the DFD Notation. The name of the warehouse is a plural noun (e.g. orders) - it derives from the input and output streams of the warehouse. The warehouse does not have to be just a data file, for example, a folder with documents, a filing cabinet, and optical discs. Therefore, viewing the warehouse in DFD is independent of implementation. The flow from the warehouse usually represents the reading of the data stored in the warehouse, and the flow to the warehouse usually expresses data entry or updating (sometimes also deleting data). Warehouse is represented by two parallel lines between which the memory name is located (it can be modelled as a UML buffer node).

3) **Data Flow**



Data flow (flow, dataflow) shows the transfer of information (sometimes also material) from one part of the system to another. The symbol of the flow is the arrow. The flow should have a name that determines what information (or what material) is being moved. Exceptions are flows where it is clear what information is transferred through the entities that are linked to these flows. Material shifts are modeled in systems that are not merely informative. Flow should only transmit one type of information (material). The arrow shows the flow direction (it can also be bi-directional if the information to/from the entity is logically dependent - e.g. question and answer). Flows link processes, warehouses and terminators.

4) **Terminator**

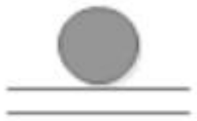


The Terminator is an external entity that communicates with the system and stands outside of the system. It can be, for example, various organizations (eg a bank), groups of people (e.g. customers), authorities (e.g. a tax office) or a department (e.g. a human-resources department) of the same organization, which does not belong to the model system. The terminator may be another system with which the modelled system communicates

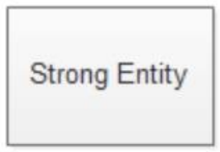
4.2 ER DIAGRAM

Entity in a database could be a single person, place, or thing about which data can be stored.

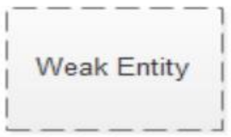
Relationship provides useful information that could not be discerned with just the entity types. Relationships are meaningful associations between or among entities like assigning, associating, or tracking.



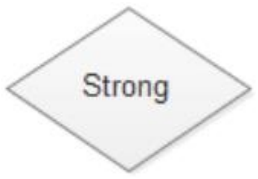
Categories are tables that are subtypes of a parent table. In relation to a database, an entity is a single person, place, or thing about which data can be stored.



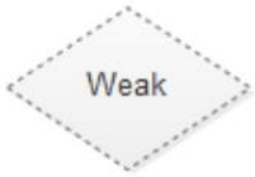
Strong entity is one that exists on its own, independent of other entities.



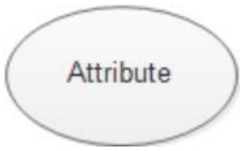
Weak entity is one whose existence depends on another entity. A weak entity cannot be uniquely identified by its attributes alone.



Strong relationships are connections that exist between a strong entity type and its owner.

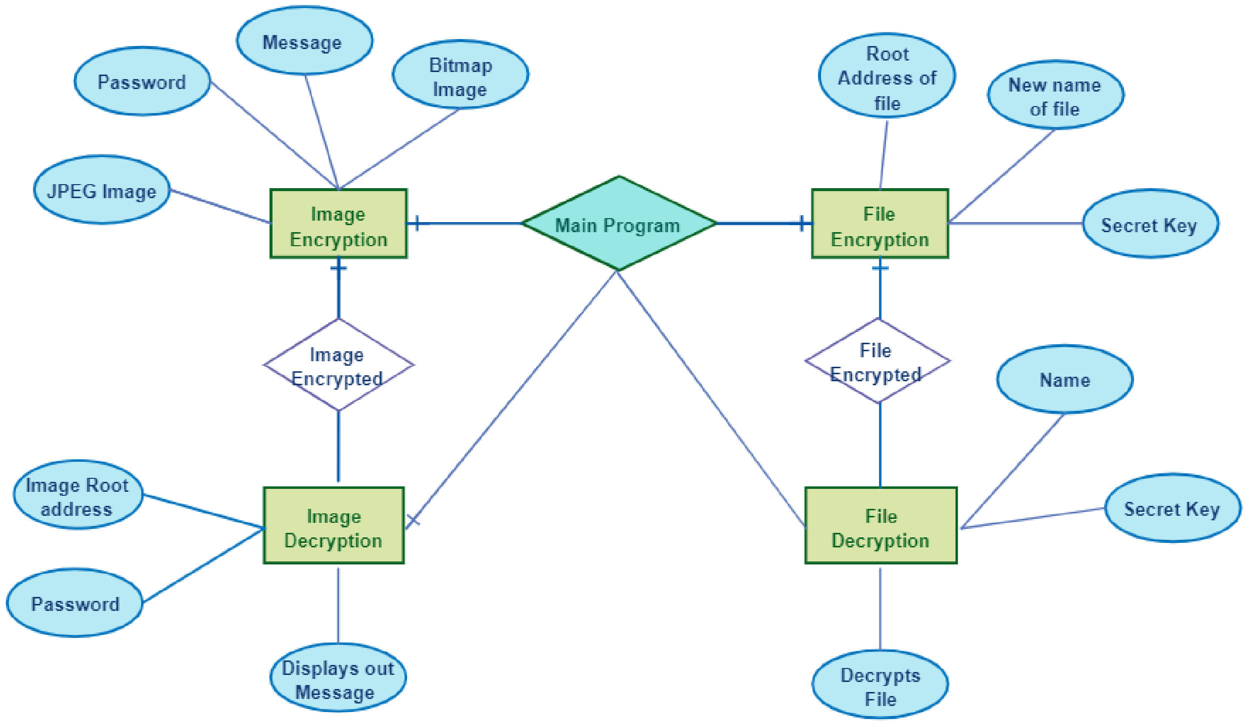


Weak relationships are connections that exist between a weak entity type and its owner.



Attribute is a piece of information which determines the properties of a field or tag in a database or a string of characters in a display.

ER DIAGRAM


Message
Password
Bitmap Image
JPEG Image
Image Encryption
Main Program
Root Address of file
New name of file
File Encryption
Secret Key
Image Encrypted
File Encrypted
Name
Image Root address
Image Decryption
File Decryption
Secret Key
Password
Displays out Message
Decrypts File

5. SYSTEM REQUIREMENTS & SPECIFICATIONS

5.1 HARDWARE REQUIREMENTS

Hardware	: Pentium IV systems or above
RAM	: 512MB (minimum)
Hard drive space	: 8GB

5.2 SOFTWARE REQUIREMENTS

Software Requirements	: Windows xp
Front End	: C# .Net

5.3 SOFTWARE SPECIFICATIONS

5.3.1 FRONT END SPECIFICATIONS

C# is a multi-paradigm programming language encompassing strong typing, imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines. It was developed by Microsoft within its .NET initiative and later approved as a standard by Ecma (ECMA-334) and ISO (ISO/IEC 23270:2006). C# is one of the programming languages designed for the Common Language Infrastructure.

.NET is built on the Windows Server System to take major advantage of the OS and which comes with a host of different servers which allows for building, deploying, managing and maintaining Webbased solutions. The Windows Server System is designed with performance as priority and it provides scalability, reliability, and manageability for the global, Web-enabled enterprise. The Windows Server System integrated software products are built for interoperability using open Web standards such as XML and SOAP.

.NET is a "Software Platform". It is a language-neutral environment for developing rich .NET experiences and building applications that can easily and securely operate within it. When developed applications are deployed, those applications will target .NET and will execute wherever .NET is implemented instead of targeting a particular Hardware/OS combination. The components that make up the .NET platform are collectively called the .NET Framework.

The .NET Framework is a managed, type-safe environment for developing and executing applications. The .NET Framework manages all aspects of program execution, like, allocation of memory for the storage of data and instructions, granting and denying permissions to the application, managing execution of the application and reallocation of memory for resources that are not needed.

The .NET Framework is designed for cross-language compatibility. Cross-language compatibility means, an application written in Visual Basic .NET may reference a DLL file written in C# (CSharp). A Visual Basic .NET class might be derived from a C# class or vice versa.

The .NET Framework consists of two main components:

• Common Language Runtime (CLR)

• Class Libraries

The advantage of C# includes

1. Powerful Windows-based Applications

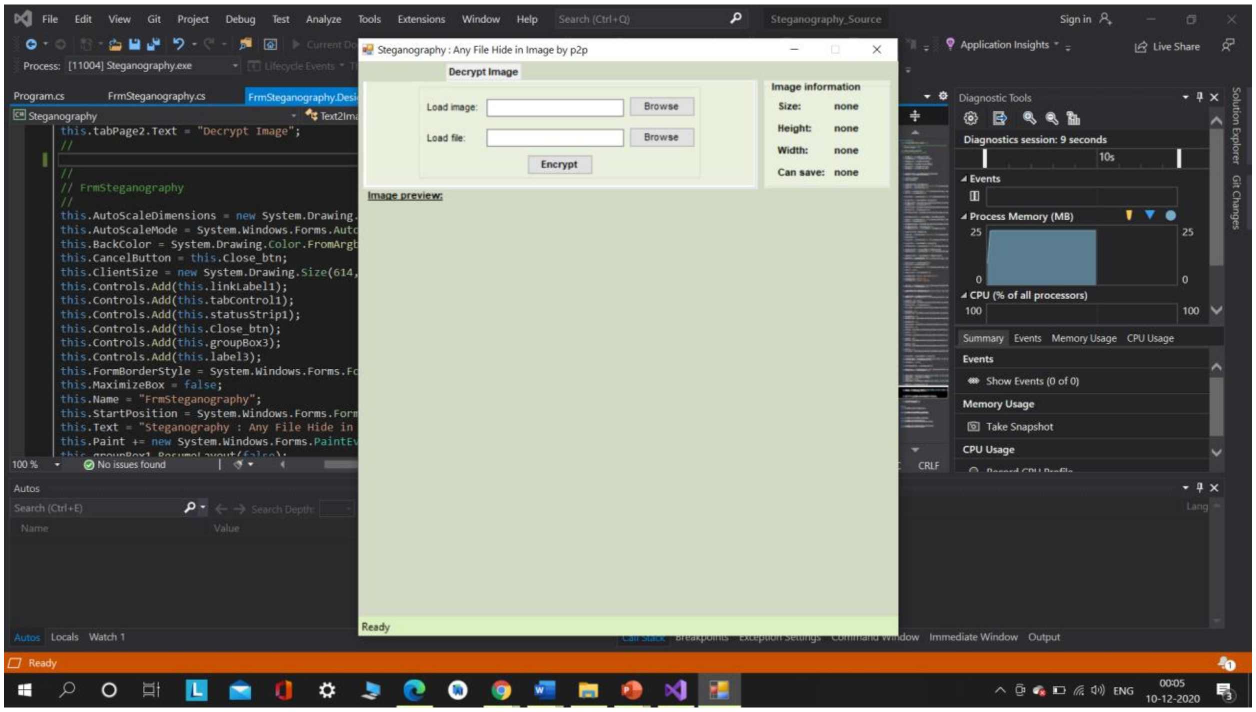
2. Building Web-based Applications

3. Simplified Deployment

- Powerful, Flexible, Simplified Data Access
- Improved Coding
- Direct Access to the Platform
- Full Object-Oriented Constructs
- XML Web Services
- COM Interoperability

6. SYSTEM DESIGN

6.1 FORM DESIGN



Process: [11004] Steganography.exe

```
this.tabPage2.Text = "Decrypt Image";
//

//
// FrmSteganography
//
this.AutoScaleDimensions = new System.Drawing.
this.AutoScaleMode = System.Windows.Forms.Auto
this.BackColor = System.Drawing.Color.FromArgb
this.CancelButton = this.Close_btn;
this.ClientSize = new System.Drawing.Size(614,
this.Controls.Add(this.linkLabel1);
this.Controls.Add(this.tabControl1);
this.Controls.Add(this.statusStrip1);
this.Controls.Add(this.Close_btn);
this.Controls.Add(this.groupBox3);
this.Controls.Add(this.label3);
this.FormBorderStyle = System.Windows.Forms.Fo
this.MaximizeBox = false;
this.Name = "FrmSteganography";
this.StartPosition = System.Windows.Forms.Form
this.Text = "Steganography : Any File Hide in
this.Paint += new System.Windows.Forms.PaintEv
```

Steganography : Any File Hide in Image by p2p

Decrypt Image

Load image: [ ] Browse

Load file: [ ] Browse

Encrypt

Image information	
Size:	none
Height:	none
Width:	none
Can save:	none

Image preview:

Ready

7. SYSTEM IMPLEMENTATION

7.1 CODING

```
using System;
using System.Drawing;
using System.Windows.Forms;
using System.IO;

namespace Text2Image
{
   public partial class FrmSteganography : Form
   {
      public FrmSteganography()
      {
         InitializeComponent();
      }

      //public values:
      string loadedTrueImagePath, loadedFilePath, saveToImage,DLoadImagePath,DSaveFilePath;
      int height, width;
      long fileSize, fileNameSize;
      Image loadedTrueImage, DecryptedImage ,AfterEncryption;
      Bitmap loadedTrueBitmap, DecryptedBitmap;
      Rectangle previewImage = new Rectangle(20,160,490,470);
      bool canPaint = false, EncriptionDone = false;
      byte[] fileContainer;

      private void EnImageBrowse_btn_Click(object sender, EventArgs e)
      {
```

```
    if (openFileDialog1.ShowDialog() == DialogResult.OK)
    {
        loadedTrueImagePath = openFileDialog1.FileName;
        EnImage_tbx.Text = loadedTrueImagePath;
        loadedTrueImage = Image.FromFile(loadedTrueImagePath);
        height = loadedTrueImage.Height;
        width = loadedTrueImage.Width;
        loadedTrueBitmap = new Bitmap(loadedTrueImage);

        FileInfo imginf = new FileInfo(loadedTrueImagePath);
        float fs = (float)imginf.Length / 1024;
        ImageSize_lbl.Text = smalldecimal(fs.ToString(), 2) + " KB";
        ImageHeight_lbl.Text = loadedTrueImage.Height.ToString() + " Pixel";
        ImageWidth_lbl.Text = loadedTrueImage.Width.ToString() + " Pixel";
        double cansave = (8.0 * ((height * (width / 3) * 3) / 3 - 1)) / 1024;
        CanSave_lbl.Text = smalldecimal(cansave.ToString(), 2) + " KB";

        canPaint = true;
        this.Invalidate();
    }
}

private string smalldecimal(string inp, int dec)
{
    int i;
    for (i = inp.Length - 1; i > 0; i--)
        if (inp[i] == '.')
            break;
    try
    {
        return inp.Substring(0, i + dec + 1);
    }
    catch
    {
        return inp;
    }
}

private void EnFileBrowse_btn_Click(object sender, EventArgs e)
{
    if (openFileDialog2.ShowDialog() == DialogResult.OK)
    {
        loadedFilePath = openFileDialog2.FileName;
        EnFile_tbx.Text = loadedFilePath;
        FileInfo finfo = new FileInfo(loadedFilePath);
        fileSize = finfo.Length;
        fileNameSize = justFName(loadedFilePath).Length;
    }
}

private void Encrypt_btn_Click(object sender, EventArgs e)
{
    if (saveFileDialog1.ShowDialog() == DialogResult.OK)
    {
        saveToImage = saveFileDialog1.FileName;
    }
    else
        return;
```

```
            if (EnImage_tbx.Text == String.Empty || EnFile_tbx.Text == String.Empty)
            {
                MessageBox.Show("Encrypton information is incomplete!\nPlease complete them frist.",
"Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            if (8*((height * (width/3)*3)/3 - 1) < fileSize + fileNameSize)
            {
                MessageBox.Show("File size is too large!\nPlease use a larger image to hide this file.", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                return;
            }
            fileContainer = File.ReadAllBytes(loadedFilePath);
            EncryptLayer();
        }



        private void EncryptLayer()
        {
            toolStripStatusLabel1.Text ="Encrypting... Please wait";
            Application.DoEvents();
            long FSize = fileSize;
            Bitmap changedBitmap = EncryptLayer(8, loadedTrueBitmap, 0, (height * (width/3)*3) / 3 -
fileNameSize - 1, true);
            FSize -= (height * (width / 3) * 3) / 3 - fileNameSize - 1;
            if(FSize > 0)
            {
                for (int i = 7; i >= 0 && FSize > 0; i--)
                {
                    changedBitmap = EncryptLayer(i, changedBitmap, (((8 - i) * height * (width / 3) * 3) / 3 -
fileNameSize - (8 - i)), (((9 - i) * height * (width / 3) * 3) / 3 - fileNameSize - (9 - i)), false);
```

```
                FSize -= (height * (width / 3) * 3) / 3 - 1;
            }
        }
        changedBitmap.Save(saveToImage);
        toolStripStatusLabel1.Text = "Encrypted image has been successfully saved.";
        EncriptionDone = true;
        AfterEncryption = Image.FromFile(saveToImage);
        this.Invalidate();
    }



    private Bitmap EncryptLayer(int layer, Bitmap inputBitmap, long startPosition, long endPosition,
bool writeFileName)
    {
        Bitmap outputBitmap = inputBitmap;
        layer--;
        int i = 0, j = 0;
        long FNSize = 0;
        bool[] t = new bool[8];
        bool[] rb = new bool[8];
        bool[] gb = new bool[8];
        bool[] bb = new bool[8];
        Color pixel = new Color();
        byte r, g, b;

        if (writeFileName)
        {
            FNSize = fileNameSize;
            string fileName = justFName(loadedFilePath);
```

```
//write fileName:
for (i = 0; i < height && i * (height / 3) < fileNameSize; i++)
  for (j = 0; j < (width / 3) * 3 && i * (height / 3) + (j / 3) < fileNameSize; j++)
  {
    byte2bool((byte)fileName[i * (height / 3) + j / 3], ref t);
    pixel = inputBitmap.GetPixel(j, i);
    r = pixel.R;
    g = pixel.G;
    b = pixel.B;
    byte2bool(r, ref rb);
    byte2bool(g, ref gb);
    byte2bool(b, ref bb);
    if (j % 3 == 0)
    {
      rb[7] = t[0];
      gb[7] = t[1];
      bb[7] = t[2];
    }
    else if (j % 3 == 1)
    {
      rb[7] = t[3];
      gb[7] = t[4];
      bb[7] = t[5];
    }
    else
    {
      rb[7] = t[6];
      gb[7] = t[7];
```

```
                }
                Color result = Color.FromArgb((int)bool2byte(rb), (int)bool2byte(gb), (int)bool2byte(bb));
                outputBitmap.SetPixel(j, i, result);
            }
        i--;
    }
    //write file (after file name):
    int tempj = j;


    for (; i < height && i * (height / 3) < endPosition - startPosition + FNSize && startPosition + i *
(height / 3) < fileSize + FNSize; i++)
        for (j = 0; j < (width / 3) * 3 && i * (height / 3) + (j / 3) < endPosition - startPosition + FNSize &&
startPosition + i * (height / 3) + (j / 3) < fileSize + FNSize; j++)
        {
            if (tempj != 0)
            {
                j = tempj;
                tempj = 0;
            }
            byte2bool((byte)fileContainer[startPosition + i * (height / 3) + j / 3 - FNSize], ref t);
            pixel = inputBitmap.GetPixel(j, i);
            r = pixel.R;
            g = pixel.G;
            b = pixel.B;
            byte2bool(r, ref rb);
            byte2bool(g, ref gb);
            byte2bool(b, ref bb);
            if (j % 3 == 0)
            {
```

```
            rb[layer] = t[0];
            gb[layer] = t[1];
            bb[layer] = t[2];
        }
        else if (j % 3 == 1)
        {
            rb[layer] = t[3];
            gb[layer] = t[4];
            bb[layer] = t[5];
        }
        else
        {
            rb[layer] = t[6];
            gb[layer] = t[7];
        }
        Color result = Color.FromArgb((int)bool2byte(rb), (int)bool2byte(gb), (int)bool2byte(bb));
        outputBitmap.SetPixel(j, i, result);

    }
long tempFS = fileSize, tempFNS = fileNameSize;
r = (byte)(tempFS % 100);
tempFS /= 100;
g = (byte)(tempFS % 100);
tempFS /= 100;
b = (byte)(tempFS % 100);
Color flenColor = Color.FromArgb(r,g,b);
outputBitmap.SetPixel(width - 1, height - 1, flenColor);

r = (byte)(tempFNS % 100);
```

```
    tempFNS /= 100;

    g = (byte)(tempFNS % 100);

    tempFNS /= 100;

    b = (byte)(tempFNS % 100);

    Color fnlenColor = Color.FromArgb(r,g,b);

    outputBitmap.SetPixel(width - 2, height - 1, fnlenColor);


    return outputBitmap;

}



private void DecryptLayer()

{

    toolStripStatusLabel1.Text = "Decrypting... Please wait";

    Application.DoEvents();

    int i, j = 0;

    bool[] t = new bool[8];

    bool[] rb = new bool[8];

    bool[] gb = new bool[8];

    bool[] bb = new bool[8];

    Color pixel = new Color();

    byte r, g, b;

    pixel = DecryptedBitmap.GetPixel(width - 1, height - 1);

    long fSize = pixel.R + pixel.G * 100 + pixel.B * 10000;

    pixel = DecryptedBitmap.GetPixel(width - 2, height - 1);

    long fNameSize = pixel.R + pixel.G * 100 + pixel.B * 10000;

    byte[] res = new byte[fSize];

    string resFName = "";

    byte temp;
```

```
//Read file name:
for (i = 0; i < height && i * (height / 3) < fNameSize; i++)
  for (j = 0; j < (width / 3) * 3 && i * (height / 3) + (j / 3) < fNameSize; j++)
  {
    pixel = DecryptedBitmap.GetPixel(j, i);
    r = pixel.R;
    g = pixel.G;
    b = pixel.B;
    byte2bool(r, ref rb);
    byte2bool(g, ref gb);
    byte2bool(b, ref bb);
    if (j % 3 == 0)
    {
      t[0] = rb[7];
      t[1] = gb[7];
      t[2] = bb[7];
    }
    else if (j % 3 == 1)
    {
      t[3] = rb[7];
      t[4] = gb[7];
      t[5] = bb[7];
    }
    else
    {
      t[6] = rb[7];
      t[7] = gb[7];
      temp = bool2byte(t);
```

```
                    resFName += (char)temp;
                }
            }

            //Read file on layer 8 (after file name):
            int tempj = j;
            i--;

            for (; i < height && i * (height / 3) < fSize + fNameSize; i++)
                for (j = 0; j < (width / 3) * 3 && i * (height / 3) + (j / 3) < (height * (width / 3) * 3) / 3 - 1 && i *
        (height / 3) + (j / 3) < fSize + fNameSize; j++)
                {
                    if (tempj != 0)
                    {
                        j = tempj;
                        tempj = 0;
                    }
                    pixel = DecryptedBitmap.GetPixel(j, i);
                    r = pixel.R;
                    g = pixel.G;
                    b = pixel.B;
                    byte2bool(r, ref rb);
                    byte2bool(g, ref gb);
                    byte2bool(b, ref bb);
                    if (j % 3 == 0)
                    {
                        t[0] = rb[7];
                        t[1] = gb[7];
                        t[2] = bb[7];
                    }
                    else if (j % 3 == 1)
                    {
                        t[3] = rb[7];
                        t[4] = gb[7];
                        t[5] = bb[7];
                    }
                    else
                    {
                        t[6] = rb[7];
                        t[7] = gb[7];
                        temp = bool2byte(t);
                        res[i * (height / 3) + j / 3 - fNameSize] = temp;
                    }
                }

            //Read file on other layers:
            long readedOnL8 = (height * (width/3)*3) /3 - fNameSize - 1;


            for (int layer = 6; layer >= 0 && readedOnL8 + (6 - layer) * ((height * (width / 3) * 3) / 3 - 1) < fSize;
        layer--)
                for (i = 0; i < height && i * (height / 3) + readedOnL8 + (6 - layer) * ((height * (width / 3) * 3) / 3
        - 1) < fSize; i++)
                    for (j = 0; j < (width / 3) * 3 && i * (height / 3) + (j / 3) + readedOnL8 + (6 - layer) * ((height *
        (width / 3) * 3) / 3 - 1) < fSize; j++)
                    {
                        pixel = DecryptedBitmap.GetPixel(j, i);
                        r = pixel.R;
                        g = pixel.G;
                        b = pixel.B;
                        byte2bool(r, ref rb);
                        byte2bool(g, ref gb);
                        byte2bool(b, ref bb);
                        if (j % 3 == 0)
                        {
                            t[0] = rb[layer];
                            t[1] = gb[layer];
                            t[2] = bb[layer];
                        }
                        else if (j % 3 == 1)
                        {
                            t[3] = rb[layer];
                            t[4] = gb[layer];
                            t[5] = bb[layer];
                        }
                        else
                        {
                            t[6] = rb[layer];
                            t[7] = gb[layer];
                            temp = bool2byte(t);
                            res[i * (height / 3) + j / 3 + (6 - layer) * ((height * (width / 3) * 3) / 3 - 1) + readedOnL8] =
        temp;
                        }
                    }

            if (File.Exists(DSaveFilePath + "\\" + resFName))
            {
                MessageBox.Show("File \"" + resFName + "\" already exist please choose another path to save
        file", "Error",MessageBoxButtons.OK,MessageBoxIcon.Error);
```

```
        return;
    }
    else
        File.WriteAllBytes(DSaveFilePath + "\\" + resFName, res);
    toolStripStatusLabel1.Text = "Decrypted file has been successfully saved.";
    Application.DoEvents();
}

private void byte2bool(byte inp, ref bool[] outp)
{
    if(inp>=0 && inp<=255)
        for (short i = 7; i >= 0; i--)
        {
            if (inp % 2 == 1)
                outp[i] = true;
            else
                outp[i] = false;
            inp /= 2;
        }
    else
        throw new Exception("Input number is illegal.");
}

private byte bool2byte(bool[] inp)
{
    byte outp = 0;
    for (short i = 7; i >= 0; i--)
    {
        if (inp[i])
```

```
                outp += (byte)Math.Pow(2.0, (double)(7-i));
            }
            return outp;
        }

        private void Decrypt_btn_Click(object sender, EventArgs e)
        {



            if (DeSaveFile_tbx.Text == String.Empty || DeLoadImage_tbx.Text == String.Empty)
            {
                MessageBox.Show("Text boxes must not be empty!", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Error);


                return;
            }

            if (System.IO.File.Exists(DeLoadImage_tbx.Text) == false)
            {
                MessageBox.Show("Select image file.", "Error", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);
                DeLoadImage_tbx.Focus();
                return;
            }




            DecryptLayer();
        }
```

```
private void DeLoadImageBrowse_btn_Click(object sender, EventArgs e)
{
  if (openFileDialog3.ShowDialog() == DialogResult.OK)
  {
    DLoadImagePath = openFileDialog3.FileName;
    DeLoadImage_tbx.Text = DLoadImagePath;
    DecryptedImage = Image.FromFile(DLoadImagePath);
    height = DecryptedImage.Height;
    width = DecryptedImage.Width;
    DecryptedBitmap = new Bitmap(DecryptedImage);

    FileInfo imginf = new FileInfo(DLoadImagePath);
    float fs = (float)imginf.Length / 1024;
    ImageSize_lbl.Text = smalldecimal(fs.ToString(), 2) + " KB";
    ImageHeight_lbl.Text = DecryptedImage.Height.ToString() + " Pixel";
    ImageWidth_lbl.Text = DecryptedImage.Width.ToString() + " Pixel";
    double cansave = (8.0 * ((height * (width / 3) * 3) / 3 - 1)) / 1024;
    CanSave_lbl.Text = smalldecimal(cansave.ToString(), 2) + " KB";

    canPaint = true;
    this.Invalidate();
  }
}

private void DeSaveFileBrowse_btn_Click(object sender, EventArgs e)
{
  if (folderBrowserDialog1.ShowDialog() == DialogResult.OK)
  {
        DSaveFilePath = folderBrowserDialog1.SelectedPath;
        DeSaveFile_tbx.Text = DSaveFilePath;
    }
}

private void Form1_Paint(object sender, PaintEventArgs e)
{
    if(canPaint)
        try
        {
            if (!EncriptionDone)
                e.Graphics.DrawImage(loadedTrueImage, previewImage);
            else
                e.Graphics.DrawImage(AfterEncryption, previewImage);
        }
        catch
        {
            e.Graphics.DrawImage(DecryptedImage, previewImage);
        }
}

private string justFName(string path)
{
    string output;
    int i;
    if (path.Length == 3)   // i.e: "C:\\"
        return path.Substring(0, 1);
    for (i = path.Length - 1; i > 0; i--)
        if (path[i] == '\\')
```

```
                break;
            output = path.Substring(i + 1);
            return output;
        }


        private string justEx(string fName)
        {
            string output;
            int i;
            for (i = fName.Length - 1; i > 0; i--)
                if (fName[i] == '.')
                    break;
            output = fName.Substring(i + 1);
            return output;
        }

        private void Close_btn_Click(object sender, EventArgs e)
        {
            this.Close();
        }



    }
}
```

```
namespace Text2Image

{

    partial class FrmSteganography

    {

        /// <summary>

        /// Required designer variable.

        /// </summary>

        private System.ComponentModel.IContainer components = null;



        /// <summary>

        /// Clean up any resources being used.

        /// </summary>

        /// <param name="disposing">true if managed resources should be disposed; otherwise,
false.</param>

        protected override void Dispose(bool disposing)

        {

            if (disposing && (components != null))

            {

                components.Dispose();

            }

            base.Dispose(disposing);

        }



        #region Windows Form Designer generated code



        /// <summary>

        /// Required method for Designer support - do not modify

        /// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.groupBox1 = new System.Windows.Forms.GroupBox();
    this.label5 = new System.Windows.Forms.Label();
    this.Decrypt_btn = new System.Windows.Forms.Button();
    this.DeLoadImage_tbx = new System.Windows.Forms.TextBox();
    this.DeSaveFile_tbx = new System.Windows.Forms.TextBox();
    this.DeSaveFileBrowse_btn = new System.Windows.Forms.Button();
    this.label6 = new System.Windows.Forms.Label();
    this.DeLoadImageBrowse_btn = new System.Windows.Forms.Button();
    this.groupBox2 = new System.Windows.Forms.GroupBox();
    this.Encrypt_btn = new System.Windows.Forms.Button();
    this.EnFileBrowse_btn = new System.Windows.Forms.Button();
    this.EnImageBrowse_btn = new System.Windows.Forms.Button();
    this.label2 = new System.Windows.Forms.Label();
    this.EnFile_tbx = new System.Windows.Forms.TextBox();
    this.label1 = new System.Windows.Forms.Label();
    this.EnImage_tbx = new System.Windows.Forms.TextBox();
    this.openFileDialog1 = new System.Windows.Forms.OpenFileDialog();
    this.openFileDialog2 = new System.Windows.Forms.OpenFileDialog();
    this.saveFileDialog1 = new System.Windows.Forms.SaveFileDialog();
    this.openFileDialog3 = new System.Windows.Forms.OpenFileDialog();
    this.label3 = new System.Windows.Forms.Label();
    this.groupBox3 = new System.Windows.Forms.GroupBox();
    this.ByteCapacity_lbl = new System.Windows.Forms.Label();
    this.CanSave_lbl = new System.Windows.Forms.Label();
    this.ImageWidth_lbl = new System.Windows.Forms.Label();
    this.ImageHeight_lbl = new System.Windows.Forms.Label();
    this.ImageSize_lbl = new System.Windows.Forms.Label();
    this.label9 = new System.Windows.Forms.Label();
    this.label10 = new System.Windows.Forms.Label();
    this.label8 = new System.Windows.Forms.Label();
    this.label7 = new System.Windows.Forms.Label();
    this.folderBrowserDialog1 = new System.Windows.Forms.FolderBrowserDialog();
    this.Close_btn = new System.Windows.Forms.Button();
    this.statusStrip1 = new System.Windows.Forms.StatusStrip();
    this.toolStripStatusLabel1 = new System.Windows.Forms.ToolStripStatusLabel();
    this.tabControl1 = new System.Windows.Forms.TabControl();
    this.tabPage1 = new System.Windows.Forms.TabPage();
    this.tabPage2 = new System.Windows.Forms.TabPage();
    this.linkLabel1 = new System.Windows.Forms.LinkLabel();
    this.groupBox1.SuspendLayout();
    this.groupBox2.SuspendLayout();
    this.groupBox3.SuspendLayout();
    this.statusStrip1.SuspendLayout();
    this.tabControl1.SuspendLayout();
    this.tabPage1.SuspendLayout();
    this.tabPage2.SuspendLayout();
    this.SuspendLayout();
    //
    // groupBox1
    //
    this.groupBox1.Controls.Add(this.label5);
    this.groupBox1.Controls.Add(this.Decrypt_btn);
    this.groupBox1.Controls.Add(this.DeLoadImage_tbx);
    this.groupBox1.Controls.Add(this.DeSaveFile_tbx);
    this.groupBox1.Controls.Add(this.DeSaveFileBrowse_btn);
```

```
            this.groupBox1.Controls.Add(this.label6);

            this.groupBox1.Controls.Add(this.DeLoadImageBrowse_btn);

            this.groupBox1.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.groupBox1.Location = new System.Drawing.Point(60, 0);

            this.groupBox1.Name = "groupBox1";

            this.groupBox1.Size = new System.Drawing.Size(320, 111);

            this.groupBox1.TabIndex = 0;

            this.groupBox1.TabStop = false;

            //

            // label5

            //

            this.label5.AutoSize = true;

            this.label5.Font = new System.Drawing.Font("Arial", 8.25F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.label5.Location = new System.Drawing.Point(7, 22);

            this.label5.Name = "label5";

            this.label5.Size = new System.Drawing.Size(65, 14);

            this.label5.TabIndex = 1;

            this.label5.Text = "Load image:";

            //

            // Decrypt_btn

            //

            this.Decrypt_btn.Font = new System.Drawing.Font("Arial", 8.25F, System.Drawing.FontStyle.Bold,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.Decrypt_btn.Location = new System.Drawing.Point(123, 82);

            this.Decrypt_btn.Name = "Decrypt_btn";

            this.Decrypt_btn.Size = new System.Drawing.Size(75, 23);

            this.Decrypt_btn.TabIndex = 3;

            this.Decrypt_btn.Text = "Decrypt";
            this.Decrypt_btn.UseVisualStyleBackColor = true;

            this.Decrypt_btn.Click += new System.EventHandler(this.Decrypt_btn_Click);

            //

            // DeLoadImage_tbx

            //

            this.DeLoadImage_tbx.Anchor =
((System.Windows.Forms.AnchorStyles)(((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Left)

                        | System.Windows.Forms.AnchorStyles.Right)));

            this.DeLoadImage_tbx.Location = new System.Drawing.Point(78, 19);

            this.DeLoadImage_tbx.Name = "DeLoadImage_tbx";

            this.DeLoadImage_tbx.Size = new System.Drawing.Size(155, 20);

            this.DeLoadImage_tbx.TabIndex = 0;

            //

            // DeSaveFile_tbx

            //

            this.DeSaveFile_tbx.Anchor =
((System.Windows.Forms.AnchorStyles)(((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Left)

                        | System.Windows.Forms.AnchorStyles.Right)));

            this.DeSaveFile_tbx.Location = new System.Drawing.Point(78, 54);

            this.DeSaveFile_tbx.Name = "DeSaveFile_tbx";

            this.DeSaveFile_tbx.Size = new System.Drawing.Size(155, 20);

            this.DeSaveFile_tbx.TabIndex = 2;

            //

            // DeSaveFileBrowse_btn

            //

            this.DeSaveFileBrowse_btn.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));
```

```
            this.DeSaveFileBrowse_btn.Font = new System.Drawing.Font("Arial", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.DeSaveFileBrowse_btn.Location = new System.Drawing.Point(239, 52);

            this.DeSaveFileBrowse_btn.Name = "DeSaveFileBrowse_btn";

            this.DeSaveFileBrowse_btn.Size = new System.Drawing.Size(75, 23);

            this.DeSaveFileBrowse_btn.TabIndex = 3;

            this.DeSaveFileBrowse_btn.Text = "Browse";

            this.DeSaveFileBrowse_btn.UseVisualStyleBackColor = true;

            this.DeSaveFileBrowse_btn.Click += new System.EventHandler(this.DeSaveFileBrowse_btn_Click);

            //

            // label6

            //

            this.label6.AutoSize = true;

            this.label6.Font = new System.Drawing.Font("Arial", 8.25F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.label6.Location = new System.Drawing.Point(7, 57);

            this.label6.Name = "label6";

            this.label6.Size = new System.Drawing.Size(64, 14);

            this.label6.TabIndex = 1;

            this.label6.Text = "Save file to:";

            //

            // DeLoadImageBrowse_btn

            //

            this.DeLoadImageBrowse_btn.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));

            this.DeLoadImageBrowse_btn.Font = new System.Drawing.Font("Arial", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.DeLoadImageBrowse_btn.Location = new System.Drawing.Point(239, 17);

            this.DeLoadImageBrowse_btn.Name = "DeLoadImageBrowse_btn";
            this.DeLoadImageBrowse_btn.Size = new System.Drawing.Size(75, 23);
            this.DeLoadImageBrowse_btn.TabIndex = 1;
            this.DeLoadImageBrowse_btn.Text = "Browse";
            this.DeLoadImageBrowse_btn.UseVisualStyleBackColor = true;
            this.DeLoadImageBrowse_btn.Click += new
System.EventHandler(this.DeLoadImageBrowse_btn_Click);
            //
            // groupBox2
            //
            this.groupBox2.Controls.Add(this.Encrypt_btn);
            this.groupBox2.Controls.Add(this.EnFileBrowse_btn);
            this.groupBox2.Controls.Add(this.EnImageBrowse_btn);
            this.groupBox2.Controls.Add(this.label2);
            this.groupBox2.Controls.Add(this.EnFile_tbx);
            this.groupBox2.Controls.Add(this.label1);
            this.groupBox2.Controls.Add(this.EnImage_tbx);
            this.groupBox2.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.groupBox2.Location = new System.Drawing.Point(60, 0);
            this.groupBox2.Name = "groupBox2";
            this.groupBox2.Size = new System.Drawing.Size(320, 111);
            this.groupBox2.TabIndex = 0;
            this.groupBox2.TabStop = false;
            //
            // Encrypt_btn
            //
            this.Encrypt_btn.Font = new System.Drawing.Font("Arial", 8.25F, System.Drawing.FontStyle.Bold,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.Encrypt_btn.Location = new System.Drawing.Point(123, 83);
            this.Encrypt_btn.Name = "Encrypt_btn";
            this.Encrypt_btn.Size = new System.Drawing.Size(75, 23);
            this.Encrypt_btn.TabIndex = 3;
            this.Encrypt_btn.Text = "Encrypt";
            this.Encrypt_btn.UseVisualStyleBackColor = true;
            this.Encrypt_btn.Click += new System.EventHandler(this.Encrypt_btn_Click);
            //
            // EnFileBrowse_btn
            //
            this.EnFileBrowse_btn.Font = new System.Drawing.Font("Arial", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.EnFileBrowse_btn.Location = new System.Drawing.Point(239, 52);
            this.EnFileBrowse_btn.Name = "EnFileBrowse_btn";
            this.EnFileBrowse_btn.Size = new System.Drawing.Size(75, 23);
            this.EnFileBrowse_btn.TabIndex = 3;
            this.EnFileBrowse_btn.Text = "Browse";
            this.EnFileBrowse_btn.UseVisualStyleBackColor = true;
            this.EnFileBrowse_btn.Click += new System.EventHandler(this.EnFileBrowse_btn_Click);
            //
            // EnImageBrowse_btn
            //
            this.EnImageBrowse_btn.Font = new System.Drawing.Font("Arial", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.EnImageBrowse_btn.Location = new System.Drawing.Point(239, 17);
            this.EnImageBrowse_btn.Name = "EnImageBrowse_btn";
            this.EnImageBrowse_btn.Size = new System.Drawing.Size(75, 23);
            this.EnImageBrowse_btn.TabIndex = 1;
            this.EnImageBrowse_btn.Text = "Browse";
            this.EnImageBrowse_btn.UseVisualStyleBackColor = true;
            this.EnImageBrowse_btn.Click += new System.EventHandler(this.EnImageBrowse_btn_Click);
            //
            // label2
            //
            this.label2.AutoSize = true;
            this.label2.Font = new System.Drawing.Font("Arial", 8.25F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label2.Location = new System.Drawing.Point(6, 57);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(51, 14);
            this.label2.TabIndex = 1;
            this.label2.Text = "Load file:";
            //
            // EnFile_tbx
            //
            this.EnFile_tbx.Location = new System.Drawing.Point(77, 54);
            this.EnFile_tbx.Name = "EnFile_tbx";
            this.EnFile_tbx.Size = new System.Drawing.Size(156, 20);
            this.EnFile_tbx.TabIndex = 2;
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Font = new System.Drawing.Font("Arial", 8.25F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label1.Location = new System.Drawing.Point(6, 22);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(65, 14);
            this.label1.TabIndex = 1;
            this.label1.Text = "Load image:";
```

```
//

// EnImage_tbx

//

this.EnImage_tbx.Location = new System.Drawing.Point(77, 20);

this.EnImage_tbx.Name = "EnImage_tbx";

this.EnImage_tbx.Size = new System.Drawing.Size(156, 20);

this.EnImage_tbx.TabIndex = 0;

//

// openFileDialog1

//

this.openFileDialog1.Filter = "Bitmap Files (*.bmp)|*.bmp|All files(*.*)|*.*";

//

// openFileDialog2

//

this.openFileDialog2.Filter = "All files (*.*)|*.*";

//

// saveFileDialog1

//

this.saveFileDialog1.Filter = "Bitmap Files (*.bmp)|*.bmp";

//

// openFileDialog3

//

this.openFileDialog3.Filter = "Bitmap Files (*.bmp)|*.bmp";

//

// label3

//

this.label3.AutoSize = true;
```

```
            this.label3.Font = new System.Drawing.Font("Arial", 8.25F,
((System.Drawing.FontStyle)((System.Drawing.FontStyle.Bold | System.Drawing.FontStyle.Underline))),
System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.label3.ForeColor = System.Drawing.Color.Black;

            this.label3.Location = new System.Drawing.Point(8, 145);

            this.label3.Name = "label3";

            this.label3.Size = new System.Drawing.Size(92, 14);

            this.label3.TabIndex = 1;

            this.label3.Text = "Image preview:";

            //

            // groupBox3

            //

            this.groupBox3.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(233)))),
((int)(((byte)(240)))), ((int)(((byte)(221)))));

            this.groupBox3.Controls.Add(this.ByteCapacity_lbl);

            this.groupBox3.Controls.Add(this.CanSave_lbl);

            this.groupBox3.Controls.Add(this.ImageWidth_lbl);

            this.groupBox3.Controls.Add(this.ImageHeight_lbl);

            this.groupBox3.Controls.Add(this.ImageSize_lbl);

            this.groupBox3.Controls.Add(this.label9);

            this.groupBox3.Controls.Add(this.label10);

            this.groupBox3.Controls.Add(this.label8);

            this.groupBox3.Controls.Add(this.label7);

            this.groupBox3.FlatStyle = System.Windows.Forms.FlatStyle.Flat;

            this.groupBox3.Font = new System.Drawing.Font("Arial", 8.25F, System.Drawing.FontStyle.Bold,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.groupBox3.ForeColor = System.Drawing.Color.Black;

            this.groupBox3.Location = new System.Drawing.Point(461, 22);

            this.groupBox3.Name = "groupBox3";

            this.groupBox3.Size = new System.Drawing.Size(144, 122);
```

```
        this.groupBox3.TabIndex = 2;

        this.groupBox3.TabStop = false;

        this.groupBox3.Text = "Image information";

        //

        // ByteCapacity_lbl

        //

        this.ByteCapacity_lbl.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));

        this.ByteCapacity_lbl.AutoSize = true;

        this.ByteCapacity_lbl.Location = new System.Drawing.Point(88, 47);

        this.ByteCapacity_lbl.Name = "ByteCapacity_lbl";

        this.ByteCapacity_lbl.Size = new System.Drawing.Size(0, 14);

        this.ByteCapacity_lbl.TabIndex = 2;

        //

        // CanSave_lbl

        //

        this.CanSave_lbl.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));

        this.CanSave_lbl.AutoSize = true;

        this.CanSave_lbl.Location = new System.Drawing.Point(77, 97);

        this.CanSave_lbl.Name = "CanSave_lbl";

        this.CanSave_lbl.Size = new System.Drawing.Size(35, 14);

        this.CanSave_lbl.TabIndex = 1;

        this.CanSave_lbl.Text = "none";

        //

        // ImageWidth_lbl

        //
```

```
            this.ImageWidth_lbl.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));

            this.ImageWidth_lbl.AutoSize = true;

            this.ImageWidth_lbl.Location = new System.Drawing.Point(77, 72);

            this.ImageWidth_lbl.Name = "ImageWidth_lbl";

            this.ImageWidth_lbl.Size = new System.Drawing.Size(35, 14);

            this.ImageWidth_lbl.TabIndex = 1;

            this.ImageWidth_lbl.Text = "none";

            //

            // ImageHeight_lbl

            //

            this.ImageHeight_lbl.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));

            this.ImageHeight_lbl.AutoSize = true;

            this.ImageHeight_lbl.Location = new System.Drawing.Point(77, 47);

            this.ImageHeight_lbl.Name = "ImageHeight_lbl";

            this.ImageHeight_lbl.Size = new System.Drawing.Size(35, 14);

            this.ImageHeight_lbl.TabIndex = 1;

            this.ImageHeight_lbl.Text = "none";

            //

            // ImageSize_lbl

            //

            this.ImageSize_lbl.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));

            this.ImageSize_lbl.AutoSize = true;

            this.ImageSize_lbl.Location = new System.Drawing.Point(77, 22);

            this.ImageSize_lbl.Name = "ImageSize_lbl";

            this.ImageSize_lbl.Size = new System.Drawing.Size(35, 14);
            this.ImageSize_lbl.TabIndex = 1;

            this.ImageSize_lbl.Text = "none";

            //

            // label9

            //

            this.label9.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));

            this.label9.AutoSize = true;

            this.label9.Location = new System.Drawing.Point(13, 97);

            this.label9.Name = "label9";

            this.label9.Size = new System.Drawing.Size(63, 14);

            this.label9.TabIndex = 0;

            this.label9.Text = "Can save: ";

            //

            // label10

            //

            this.label10.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));

            this.label10.AutoSize = true;

            this.label10.Location = new System.Drawing.Point(13, 72);

            this.label10.Name = "label10";

            this.label10.Size = new System.Drawing.Size(44, 14);

            this.label10.TabIndex = 0;

            this.label10.Text = "Width: ";

            //

            // label8

            //
            this.label8.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));

            this.label8.AutoSize = true;

            this.label8.Location = new System.Drawing.Point(13, 47);

            this.label8.Name = "label8";

            this.label8.Size = new System.Drawing.Size(48, 14);

            this.label8.TabIndex = 0;

            this.label8.Text = "Height: ";

            //

            // label7

            //

            this.label7.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Right)));

            this.label7.AutoSize = true;

            this.label7.Location = new System.Drawing.Point(13, 22);

            this.label7.Name = "label7";

            this.label7.Size = new System.Drawing.Size(36, 14);

            this.label7.TabIndex = 0;

            this.label7.Text = "Size: ";

            //

            // Close_btn

            //

            this.Close_btn.Anchor =
((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom |
System.Windows.Forms.AnchorStyles.Right)));

            this.Close_btn.DialogResult = System.Windows.Forms.DialogResult.Cancel;

            this.Close_btn.Location = new System.Drawing.Point(527, 604);

            this.Close_btn.Name = "Close_btn";

            this.Close_btn.Size = new System.Drawing.Size(75, 23);
            this.Close_btn.TabIndex = 3;
            this.Close_btn.Text = "Close";
            this.Close_btn.UseVisualStyleBackColor = true;
            this.Close_btn.Visible = false;
            this.Close_btn.Click += new System.EventHandler(this.Close_btn_Click);
            //
            // statusStrip1
            //
            this.statusStrip1.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(221)))),
        ((int)(((byte)(240)))), ((int)(((byte)(192)))));
            this.statusStrip1.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
            this.toolStripStatusLabel1});
            this.statusStrip1.Location = new System.Drawing.Point(0, 630);
            this.statusStrip1.Name = "statusStrip1";
            this.statusStrip1.Size = new System.Drawing.Size(614, 22);
            this.statusStrip1.SizingGrip = false;
            this.statusStrip1.TabIndex = 4;
            this.statusStrip1.Text = "statusStrip1";
            //
            // toolStripStatusLabel1
            //
            this.toolStripStatusLabel1.Name = "toolStripStatusLabel1";
            this.toolStripStatusLabel1.Size = new System.Drawing.Size(38, 17);
            this.toolStripStatusLabel1.Text = "Ready";
            //
            // tabControl1
            //
            this.tabControl1.Controls.Add(this.tabPage1);
            this.tabControl1.Controls.Add(this.tabPage2);
            this.tabControl1.Font = new System.Drawing.Font("Arial", 8.25F, System.Drawing.FontStyle.Bold,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.tabControl1.ImeMode = System.Windows.Forms.ImeMode.NoControl;

            this.tabControl1.Location = new System.Drawing.Point(5, 0);

            this.tabControl1.Multiline = true;

            this.tabControl1.Name = "tabControl1";

            this.tabControl1.SelectedIndex = 0;

            this.tabControl1.Size = new System.Drawing.Size(449, 145);

            this.tabControl1.TabIndex = 5;

            //

            // tabPage1

            //

            this.tabPage1.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(233)))),
((int)(((byte)(240)))), ((int)(((byte)(221)))));

            this.tabPage1.Controls.Add(this.groupBox2);

            this.tabPage1.Location = new System.Drawing.Point(4, 23);

            this.tabPage1.Name = "tabPage1";

            this.tabPage1.Padding = new System.Windows.Forms.Padding(3);

            this.tabPage1.Size = new System.Drawing.Size(441, 118);

            this.tabPage1.TabIndex = 0;

            this.tabPage1.Text = "Encrypt Image";

            //

            // tabPage2

            //

            this.tabPage2.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(233)))),
((int)(((byte)(240)))), ((int)(((byte)(221)))));

            this.tabPage2.Controls.Add(this.groupBox1);

            this.tabPage2.Location = new System.Drawing.Point(4, 23);

            this.tabPage2.Name = "tabPage2";

            this.tabPage2.Padding = new System.Windows.Forms.Padding(3);
            this.tabPage2.Size = new System.Drawing.Size(441, 118);

            this.tabPage2.TabIndex = 1;

            this.tabPage2.Text = "Decrypt Image";

            //


            //

            // FrmSteganography

            //

            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

            this.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(210)))), ((int)(((byte)(218)))),
((int)(((byte)(196)))));

            this.CancelButton = this.Close_btn;

            this.ClientSize = new System.Drawing.Size(614, 652);

            this.Controls.Add(this.linkLabel1);

            this.Controls.Add(this.tabControl1);

            this.Controls.Add(this.statusStrip1);

            this.Controls.Add(this.Close_btn);

            this.Controls.Add(this.groupBox3);

            this.Controls.Add(this.label3);

            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedSingle;

            this.MaximizeBox = false;

            this.Name = "FrmSteganography";

            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;

            this.Text = "Steganography : Any File Hide in Image by p2p";

            this.Paint += new System.Windows.Forms.PaintEventHandler(this.Form1_Paint);

            this.groupBox1.ResumeLayout(false);

            this.groupBox1.PerformLayout();

            this.groupBox2.ResumeLayout(false);
```

```
    this.groupBox2.PerformLayout();
    this.groupBox3.ResumeLayout(false);
    this.groupBox3.PerformLayout();
    this.statusStrip1.ResumeLayout(false);
    this.statusStrip1.PerformLayout();
    this.tabControl1.ResumeLayout(false);
    this.tabPage1.ResumeLayout(false);
    this.tabPage2.ResumeLayout(false);
    this.ResumeLayout(false);
    this.PerformLayout();

}

#endregion

private System.Windows.Forms.GroupBox groupBox1;
private System.Windows.Forms.GroupBox groupBox2;
private System.Windows.Forms.Button EnImageBrowse_btn;
private System.Windows.Forms.Label label1;
private System.Windows.Forms.TextBox EnImage_tbx;
private System.Windows.Forms.Button EnFileBrowse_btn;
private System.Windows.Forms.Label label2;
private System.Windows.Forms.TextBox EnFile_tbx;
private System.Windows.Forms.Button Encrypt_btn;
private System.Windows.Forms.OpenFileDialog openFileDialog1;
private System.Windows.Forms.OpenFileDialog openFileDialog2;
private System.Windows.Forms.SaveFileDialog saveFileDialog1;
private System.Windows.Forms.Label label5;
private System.Windows.Forms.Button Decrypt_btn;
```

```
        private System.Windows.Forms.TextBox DeLoadImage_tbx;
        private System.Windows.Forms.TextBox DeSaveFile_tbx;
        private System.Windows.Forms.Button DeSaveFileBrowse_btn;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.Button DeLoadImageBrowse_btn;
        private System.Windows.Forms.OpenFileDialog openFileDialog3;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.GroupBox groupBox3;
        private System.Windows.Forms.Label label7;
        private System.Windows.Forms.Label label9;
        private System.Windows.Forms.Label ByteCapacity_lbl;
        private System.Windows.Forms.Label ImageSize_lbl;
        private System.Windows.Forms.FolderBrowserDialog folderBrowserDialog1;
        private System.Windows.Forms.Label CanSave_lbl;
        private System.Windows.Forms.Label ImageWidth_lbl;
        private System.Windows.Forms.Label ImageHeight_lbl;
        private System.Windows.Forms.Label label10;
        private System.Windows.Forms.Label label8;
        private System.Windows.Forms.Button Close_btn;
        private System.Windows.Forms.StatusStrip statusStrip1;
        private System.Windows.Forms.ToolStripStatusLabel toolStripStatusLabel1;
        private System.Windows.Forms.TabControl tabControl1;
        private System.Windows.Forms.TabPage tabPage1;
        private System.Windows.Forms.TabPage tabPage2;
        private System.Windows.Forms.LinkLabel linkLabel1;
    }
}
```

8. TESTING

Software testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves execution of a software component or system component to evaluate one or more properties of interest. Software testing also helps to identify errors, gaps or missing requirements in contrary to the actual requirements. It can be either done manually or using automated tools. Some prefer saying Software testing as a White Box and Black Box Testing. In simple terms, Software Testing means Verification of Application Under Test (AUT).

8.1 OBJECTIVES

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. It is the major quality measure employed during software development.

Testing is the exposure of the system to trial input to see whether it produces correct output. It is a process, which reveals errors in the program. During testing, the program is executed with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as it is expected to perform.

8.2 TEST PROCESS

TEST STRATEGY AND TEST PLAN

Every project needs a Test Strategy and a Test Plan. These artefacts describe the scope for testing for a project:

1. The systems that need to be tested, and any specific configurations
2. Features and functions that are the focus of the project
3. Non-functional requirements
4. Test approach—traditional, exploratory, automation, etc.—or a mix
5. Key processes to follow – for defects resolution, defects triage
6. Tools—for logging defects, for test case scripting, for traceability
7. Documentation to refer, and to produce as output

8. Test environment requirements and setup
9. Risks, dependencies and contingencies
10. Test Schedule
11. Approval workflows
12. Entry/Exit criteria

And so on… Whatever methodology your project follows, you need to have a Test Strategy and Software Testing Plan in place. Make them two separate documents, or merge them into one.

Without a clear test strategy and a detailed test plan, even Agile projects will find it difficult to be productive. Why, you ask? Well, the act of creating a strategy and plan bring out a number of dependencies that you may not think of otherwise.

TEST DESIGN

Test design as a process is an amalgamation of the Test Manager's experience of similar projects over the years, testers' knowledge of the system/functionality being tested and prevailing practices in testing at any given point. For instance, if you work for a company in the early stages of a new product development, your focus will be on uncovering major bugs with the alpha/beta versions of your software, and less on making the software completely bug-proof.

TEST EXECUTION

Test execution can done in many different ways—as single, waterfall SIT (System Integration Test) and UAT (User Acceptance Test) phases; as part of Agile sprints; supplemented with exploratory tests; or with test-driven development. There is a need to do adequate amount of software testing to ensure your system is (relatively) bug-free.

TEST CLOSURE

Right—so you have done the planning necessary, executed tests and now want to green-light your product for release. You need to consider the exit criteria for signalling completion of the test cycle and readiness for a release. Let's look at the components of exit criteria in general:

1. 100% requirements coverage: all business and technical requirements have to be covered by testing.
2. Minimum % pass rate: targeting 90% of all test cases to be passed is best practice.

3. All critical defects to be fixed: self-explanatory. They are critical for a reason.

8.3 TEST CASES

A test case is a set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly. The process of developing test cases can also help find problems in the requirements or design of an application.

8.4 TESTING TYPES

UNIT TESTING

Unit testing is essentially for the verification of the code produced during the coding phase and the goal is test the internal logic of the module/program.

INTEGRATION TESTING

All the tested modules are combined into sub systems, which are then tested. The goal is to see if the modules are properly integrated, and the emphasis being on the testing interfaces between the modules.

SYSTEM TESTING

It is mainly used if the software meets its requirements. The reference document for this process is the requirement document.

ACCEPTANCE TESTING

It is performed with realistic data of the client to demonstrate that the software is working satisfactorily.

WHITE BOX TESTING

It is a test case design method that uses the control structures of the procedural design to derive test cases. In this the test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases.

Using this testing a Software Engineer can derive the following test cases: Exercise all the logical decisions on either true or false sides. Execute all loops at their boundaries and within their operational boundaries. Exercise the internal data structures to assure their validity. White Box testing attempts to find errors in the following categories:

- Guarantee that all independent paths have been executed.
- Execute all logical decisions on their true and false Sides.
- Execute all loops at their boundaries and within their operational bounds
- Execute internal data structures to ensure their validity.

BLACK BOX TESTING

It is a test case design method used on the functional requirements of the software. In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program It will help a software engineer to derive sets of input conditions that will exercise all the functional requirements of the program.

Black Box testing attempts to find errors in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structure or external database access 38
- Performance errors
- Initialization and termination errors

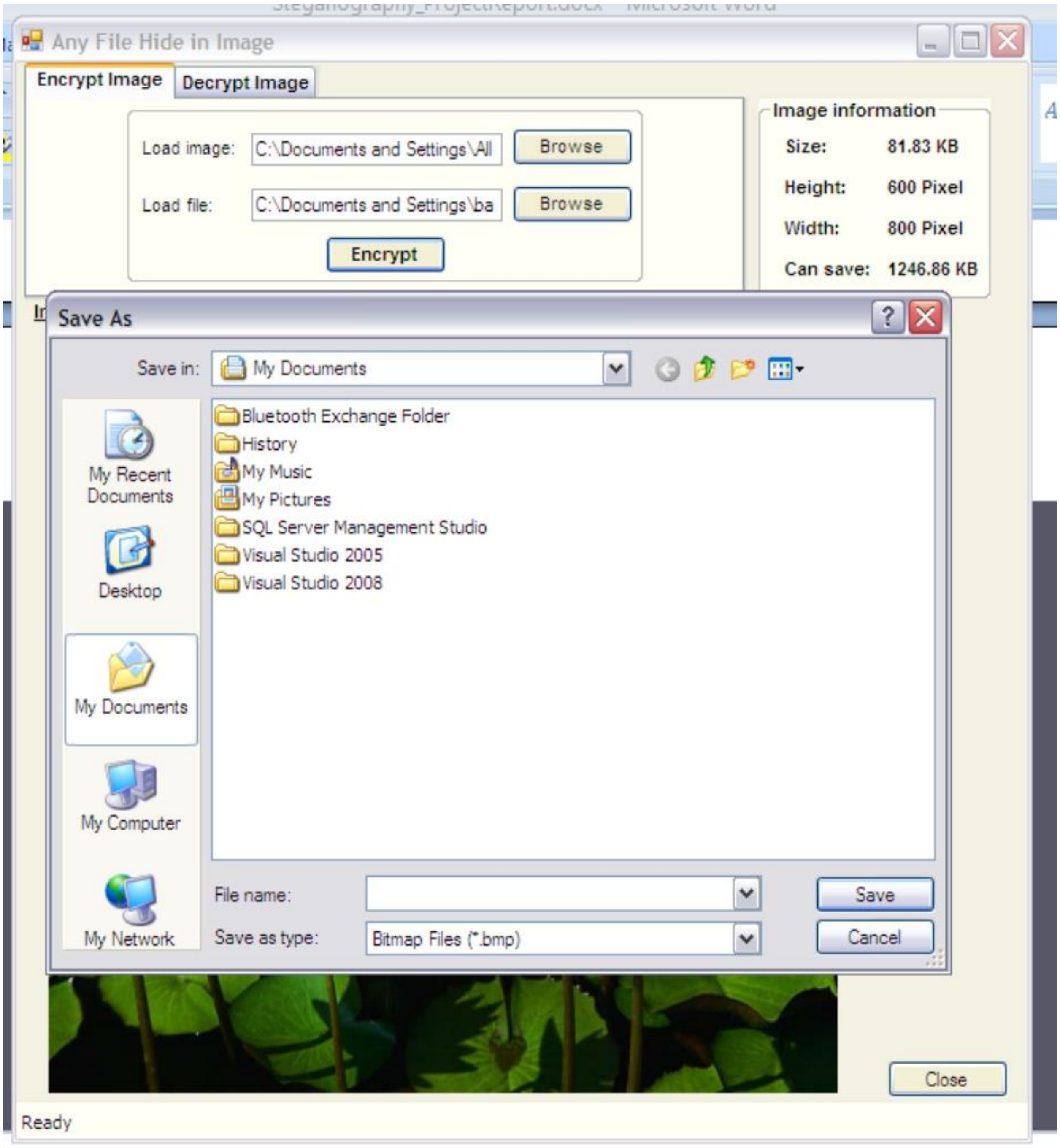
TEST RESULTS

All the test cases mentioned above passed successfully. No defects encountered.

8.5 VERIFICATION AND VALIDATION

The system has been tested and implemented successfully and thus ensured that all the requirements as listed in the software requirements specification are completely fulfilled. In case of erroneous input corresponding error messages are displayed.

9. REPORT

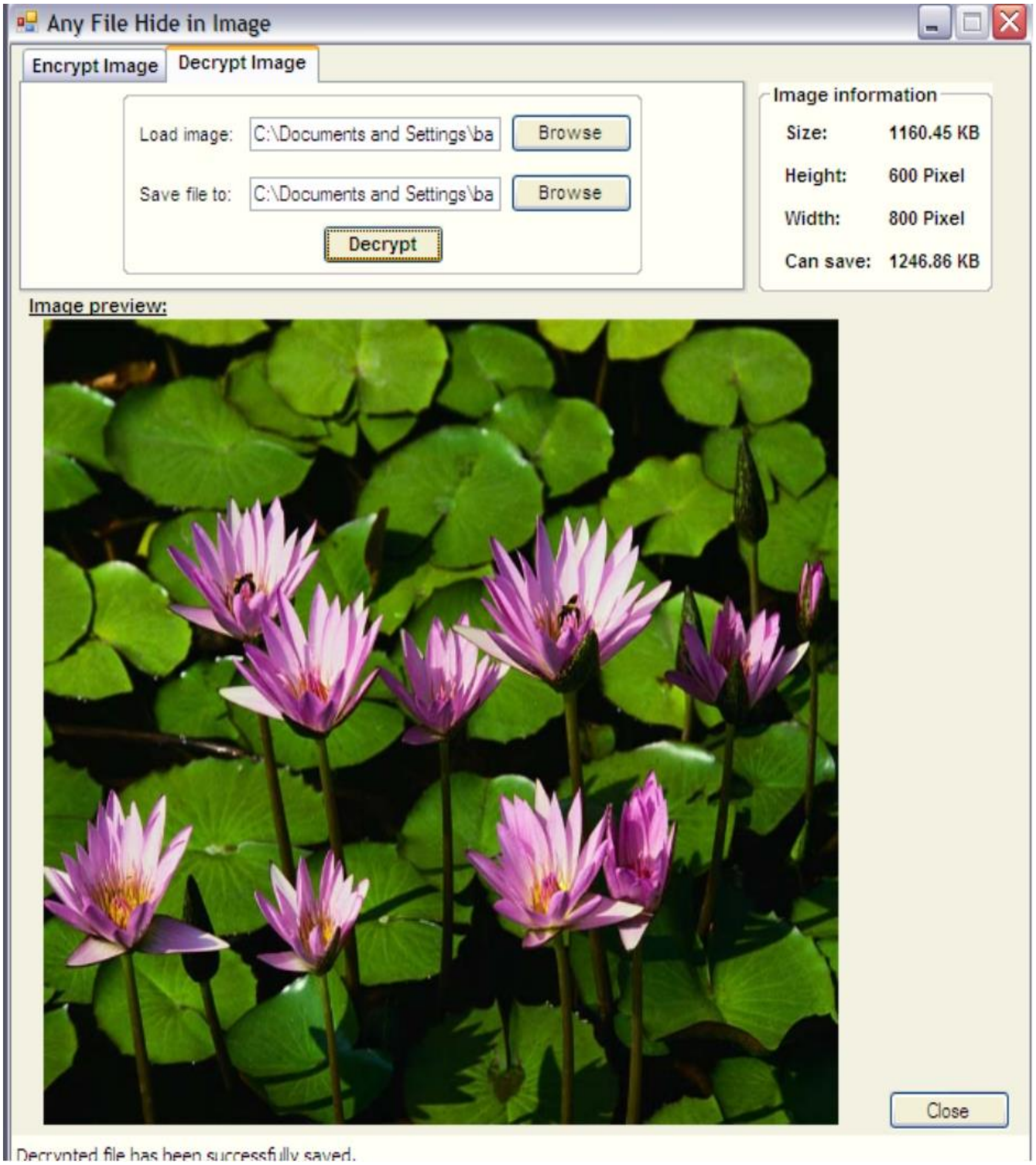


Load image: C:\Documents and Settings\All
Load file: C:\Documents and Settings\ba

Size:	81.83 KB
Height:	600 Pixel
Width:	800 Pixel
Can save:	1246.86 KB

Bluetooth Exchange Folder
History
My Music
My Pictures
SQL Server Management Studio
Visual Studio 2005
Visual Studio 2008

Save as type: Bitmap Files (*.bmp)



Load image: C:\Documents and Settings\ba

Save file to: C:\Documents and Settings\ba

Image information

Size: 1160.45 KB

Height: 600 Pixel

Width: 800 Pixel

Can save: 1246.86 KB

Decrypted file has been successfully saved.

CONCLUSION

Steganography is a really interesting subject and outside of the mainstream cryptography and system administration that most of us deal with day after day.

Steganography can be used for hidden communication. We have explored the limits of steganography theory and practice. We printed out the enhancement of the image steganography system using LSB approach to provide a means of secure communication. A stego-key has been applied to the system during embedment of the message into the cover image.

This steganography application software provided for the purpose to how to use any type of image formats to hiding any type of files inside their. The master work of this application is in supporting any type of pictures without need to convert to bitmap, and lower limitation on file size to hide, because of using maximum memory space in pictures to hide the file.

Since ancient times, man has found a desire in the ability to communicate covertly. The recent explosion of research in watermarking to protect intellectual property is evidence that steganography is not just limited to military or espionage applications. Steganography, like cryptography, will play an increasing role in the future of secure communication in the "digital world".

FUTURE ENHANCEMENT

To make it pure steganography application.

REFERENCES

• Professional C#, 2nd Edition, Simon Robinson, K. Scott Allen, Ollie Cornes, Jay Glynn, Zach Greenvoss, Burton Harvey, Christian Nagel, Morgan Skinner, Karli Watson, ISBN: 978-0- 7645-4398-2 .

• Professional ASP.NET, Christian Wenz, Jason N. Gaylord, Pranav Rastogi,Scott Hanselman, Todd Miranda, ISBN 978-1-118-31182-0, 2013

• MCAD/MCSD Self-Paced Training Kit: Developing Web Applications with Microsoft® Visual Basic® .NET and Microsoft Visual C#® .NET, 2013, Second Edition

• C# 6.0 in a Nutshell: The Definitive Reference 6th Edition, 2015, O'Reilly Media, Ben Albahari, Joseph Albahari.