

# Stego-Encrypter

## Final Project Report

Group Number-03

### Team

C. Vishwesh	140050031
Yathansh Kathuria	140050021
Rahul Kiran	140050071

# CONTENTS

- **Introduction**

1. Purpose
2. Scope
3. Overview

- **Overall Description**

1. Input data
2. The key
3. Specific Requirements
4. Functions used and their Algorithms

- **Work Distribution**

1. C. Vishwesh
2. Yathansh Kathuria
3. Rahul Kiran

# INTRODUCTION

- **Purpose:**

The application EncryptoSteganography has been developed to hide text messages (after encryption) into a image file using a key (which would be just another alpha numeric string).

This document present a detailed description of the Stego-Encrypter application. It will explain the purpose and features of the application, the interfaces of the application, what the application will do and how the application will react to external stimuli.

- **Scope:**

This application software is a Steganography application which includes encryption of the messages to be hidden as well. This application is designed to hide a text message effectively in a bitmap image using the steganography techniques (LSB). The application will also implement encryption of the text message to be hidden before it's passed into the image thus increasing the security of the hidden message many folds. This application is basically designed to allow the user to hide secret messages into a "bmp" file after its encrypted using a key as wished by a user. The aim of the application is to make the "post-processed" bmp file (after the message has been inserted, also called stego-image) look more and more identical to the original image so that it arouses least suspicion (that some hidden message is there in the image).

This software has great implemntation in various fields ranging from local communication to high security data transfer.

- **Overview:**

This application software is a Steganography application which includes encryption of the messages to be hidden as well. This application will be designed to hide a text message effectively in a bitmap image file using the steganography techniques. The application will also implement encryption of the text message to be hidden before it's passed into the image thus increasing the security of the hidden message many folds. This application is basically designed to allow the user to hide secret messages into a image file after it is encrypted using a key as wished by a user.

# OVERALL DESCRIPTION

- **Input Data:**

Our application Encrypto-Steganography has been developed to hide text messages (after encryption) into a bmp file image. So the input method are as follows:

**1.For Encryption:** The name of text file, image file and the key.

**2.For Decryption:** The name of input image,output text file and the key.

- **The Key:**

The application needs a key to encode and decode data. The text will be encrypted and decrypted on the basis of a key.

Our applicatuion is based on a symmetric key i.e. the person who is decrypting the file must use the same key as the one who encoded it to get the correct text file back. The symmetric key is the most common method of setting a passcode and is the most user friendly to operate.

- **Specific Requirements:**

In our code of steganography, the library CImg was used to load the image file and read various pixel color values so as to modify their least significant bit value and implement steganography. This library was easy to use and integrate with code blocks. It was a collection of header files which could all be accessed through the code `#include<CImg.h>`.

For the Graphic User Interface we used GTKMM, a specially designed graphiclibrary for C++. It allowed us to make windows, buttons, entry fields and labels for our User Friendly GUI.

To make a long storyshort we used the following header file and libraries:

1. iostream
2. fstream
3. cstdio
4. string
5. cmath
6. Cimg
7. gtkmm

- **Functions Used And Their Algorithms:**

- 1. Function for encryption:**

- This function will be used to encrypt data (text file). Our cipher for encryption has been inspired by the Advanced Encryption Standard (AES), a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001. It basically requires a key to encrypt data. This function will use a three step process to encrypt data:

- The function will read the data from a text file and changes all the letters on the basis of a predefined look up table (Character with ASCII code  $x$  will be replaced with a character with ASCII code  $255-x$ ).
    - Next the function divides the already ciphered text into strings of length equal to the length of the key and adds the letters of key (i.e. the ASCII code of the key is added to the data) to each block.

- Eg: The text is: bat.

- The key is: bat.

- So the ciphered text would be: (ASCII 196)(ASCII 194)(ASCII 232).

- After doing so we will store the text in a  $n \times n$  2D array and periodically shift the columns to left (number of places shifted = row number) .

- 2. Function For Decryption:**

- This function will be used to decrypt the encoded text file. The steps followed by it would be just the opposite of those followed for encryption i.e. will be in reverse order.

### **3. Functions for Steganography encryption:**

There are currently three effective methods in applying Image Steganography: LSB Substitution, Blocking, and Palette Modification. LSB (Least Significant Bit) Substitution is the process of modifying the least significant bit of the pixels of the carrier image. We have chosen to implement LSB Substitution in our project because of its simplicity as well as its efficiency. LSB Substitution works by iterating through the pixels of an image and extracting the ARGB values. It then separates the color channels and gets the least significant bit. We have used the Cimg library for editing the .bmp image files.

In this function we use 3 pixels of the image to store each character of the data. We modified the Least Significant Bit of each colour element (RGB) of each pixel to store our data. Changing the LSB hardly causes any change in the colour contrast, and the small changes caused cannot be detected by the normal human eye, thus for us the image looks to be unedited.

### **4. Function for Steganography Decryption:**

The function will be used to obtain ciphered text from the image for further processing by the decrypt function. This function will go through the least significant bits of the pixels to obtain the data back.

### **5. Graphic User Interface:**

We used GTKMM to make our Graphic User Interface (GUI). GTKMM is a graphic library specially designed for C++ programming.

Our Application starts with a home page asking whether the user wants to encrypt a file or decrypt it.

The Encrypt window has spaces for inputting the input file name, the output file name, the image file name and the key. It has two encryption options: normal encryption of file and stego encryption of file.

The decrypt window again has spaces for inputting the input image file OR the input text file, the output text file and the key. It also has two options for decryption: One from image and the other from textfile.

# WORK DISTRIBUTION

**1. C. Vishwesh:**

Code for steganography encrypt and decrypt, and correction of Decrypt function.

**2. Yathansh Kathuria:**

Code for encrypt function and decrypt function, GUI(gtkmm) and integration of different functions.

**3. Rahul Kiran:**

All documentation

This was a brief division of work. The following pages consist of detailed explanation of work done by all the members of the team

### **1. C. Vishwesh:**

I was the group leader of our team and is the one responsible behind the whole code for Steganography. I used the Cimg library for image handling. For this project I had to extract all the pixel values and then had to modify them to hide our ciphered data. I used a systematic way in which some initial pixels of the image were reserved to store the file size and the rest to hide the file itself. I modified the LSB of all the three colour pixels (RGB) of the image to hide our data in an effective way.

I also edited and corrected the decrypt function made by Rahul Kiran so that it worked properly and in accordance with the encrypt function.

### **2. Yathansh Kathuria:**

I worked on the encryption techniques and made the code for encrypting data and Decrypting data. The encryption function is a three step procedure which makes use of a key to encrypt data for high security whereas the decrypt function was just the opposite of it. The elaborate mechanism of the two is described earlier.

I also made the GUI using gtkmm and integrated all the functions so that they work in collaboration with each other i.e. I made the main function for the program.

### **3. Rahul Kiran:**

I did not know much about coding much before this project so I just tried to learn from the project and helped Yathansh and Vishwesh a bit in making their functions.

I also made the whole documentation for the program.

# References

1. <http://en.wikipedia.org/wiki/Steganography>
2. <http://en.wikipedia.org/wiki/Encryption>
3. [http://en.wikipedia.org/wiki/BMP\\_file\\_format](http://en.wikipedia.org/wiki/BMP_file_format)
4. <http://CImg.sourceforge.net/>
5. <https://developer.gnome.org/gtkmm/>
6. <http://blog.mpshouse.com/?p=884>
7. Stackoverflow.com
8. cplusplus.com
9. google.com