

# ENCRYPTO- STEGANOGRAPHY

Final Project report

**(Modified)  
(Software Requirements)  
(Specification)**

Lab batch: 362

Sudipto Biswas	110050048
Suman Naskar	115060018
Sudheer Jhajharia	110040043
TLD Manoj	110050063
Subrat Sahoo	115060003
Surya Kr Singh	115320008

# CONTENTS

1. Introduction
  - 1.1 Purpose
  - 1.2 Scope
  - 1.3 Glossary
  - 1.4 Overview
2. Overall Description
  - 2.1 Input Data
  - 2.2 Description of Algorithm to be used
  - 2.3 Output Data
3. Technologies to be used
4. Specific Requirements
5. Work Division and explanation of individual contribution
  - 5.1 Sudipto Biswas
  - 5.2 Sudheer
  - 5.3 TLD Manoj
  - 5.4 Surya Kr Singh
  - 5.5 Sumon Naskar
  - 5.6 Subhrat Kr Sahoo
6. Self-Evaluation (Peer reviewed)
7. References

# 1.INTRODUCTION

## 1.1 Purpose

The purpose of this document is to present a detailed description of the Encrypto-Steganography application. It will explain the purpose and features of the application, the interfaces of the application, what the application will do, the constraints under which it must operate and how the application will react to external stimuli. This document is intended for both the stakeholders and the developers of the system.

## 1.2 Scope

This application software will be 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 (known widely as “bmp” files) 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 “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). Steganography has been used since ancient times and now-a-days large corporations use the modern digital steganography to hide information that they send from say one industry to another. Thus steganography including encryption is immensely popular in today’s world for hiding secure information.

## 1.3 Glossary

Term	Definition
GUI	Graphical User Interface.
Software Requirements Specification	A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. For example, this document.
Steganography	Steganography is the art and science of writing hidden messages in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message, a form of security through obscurity.
Encryption	Encryption is the process of transforming information (referred to as plaintext) using an

	algorithm (called cipher) to make it unreadable to anyone except those possessing special knowledge, usually referred to as a key.
Library	A library is a collection of resources used to develop software. These may include pre-written code and subroutines, classes, values or type specifications.

## 1.4 Overview

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.

The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product. Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

# 2. OVERALL DESCRIPTION

## 2.1 Input Data

Our application Encrypto-Steganography has been developed to hide text messages (after encryption) into a bmp file image. So as input it will require the full name of the bmp file (along with extension), the text message that is to be hidden and the key (basically another sequence of characters, can be a word or numbers etc.) based on which the text message will be encrypted.

## 2.2 Description of Algorithm used

### 1. Algorithm for Steganography code:

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 (here bmp file) and extracting the ARGB values. It then separates the color channels and gets the least significant bit.

Meanwhile, it also iterates through the characters of the message replacing each of least significant bits(i.e. the last of the 8 bits of a byte) in every byte of the image by every bit of the text message. This is implemented by including the well-known C library called “EasyBMP” and application of bitwise operators(<<,&,&| etc.).

## 2. Algorithm for Encryption code:

In case of our encryption code the following two ciphers(methods of encryption) will be used: Substitution cipher and Transposition cipher. In **substitution cipher** the units of plaintext are replaced with cipher text according to a regular system; the "units" may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth. The receiver deciphers the text by performing an inverse substitution. There are also many types of substitution and transposition ciphers but in our code we will be using only simple substitution and columnar transposition.

### 2.3 Output Data

The encrypted text message after being inserted bit by bit in the LSBs of the bytes of the image will yield the desired output of what is known as the stego-image. The success of the code lies in how much the original and the stego-image are identical. The stego-image that is produced will definitely be different from the original image but the changes will be far small to be conceived by the human eye, thus securing the message.

## 3.SPECIFIC REQUIREMENTS

In the code of steganography, the library EasyBMP (version 1.06) will be used to load the bmp file and read various pixel color values so as to modify their least significant bit value. The EasyBMP library has been taken from the link: <http://easybmp.sourceforge.net/download.html> This is a compressed zip file which consists of a few header files and a C++ code for EasyBMP. Installing the EasyBMP library is easy. We just simply copy all the \*.h and \*.cpp files to the directory of our project. Alternatively, we copy all the header files anywhere in our compiler's path. We should have the following files:

1. EasyBMP.h
2. EasyBMP DataStructures.h
3. EasyBMP BMP.h
4. EasyBMP VariousBMPutilities.h
5. EasyBMP.cpp

along with a code sample in the sample directory. To use the EasyBMP library, simply include the EasyBMP.h file via

```
#include "EasyBMP.h"
```

Note that if you have copied all the EasyBMP source files to your compiler path, you may not need the quotes, but rather brackets:

```
#include <EasyBMP.h>
```

To access various pixel values in a bmp file and to modify them we require certain functions which are provided by the EasyBMP library. For example to load an BMP image we create an BMP file type object and then load the BMP file onto which the message will be hidden in this object. This is done as following:

```
BMP AnImage;  
AnImage.ReadFromFile("sample.bmp");  
cout << "File info:" << endl;  
cout << AnImage.TellWidth() << " x " << AnImage.TellHeight()<< " at " <<  
AnImage.TellBitDepth() << " bpp" << endl;
```

The above code declares a BMP class object AnImage and then loads the bmp file sample.bmp into it. Similarly the functions TellWidth(), TellHeight() which are the member functions of the class BMP tell the size of the image in terms of number of pixels. Similarly there are a large number of other functions which will be used in the main code for steganography.

Finally we compile the source code as we normally would along with the single EasyBMP.cpp file. For instance, to compile the code for steganography (say its name is stego.cpp) with g++, we use:

```
g++ -o stego stego.cpp EasyBMP.cpp
```

## 4. TECHNOLOGIES TO BE USED

The basic programming language in which the application code is written is C/C++. C++ (pronounced "cee plus plus") is a statically typed, free-form, multi-paradigm, compiled, general-purpose programming language. It is regarded as an intermediate-level language, as it comprises a combination of both high-level and low-level language features. The code for steganography as well as encryption has been written in C++. The GUI for this Encrypto-Steganography application has been created using the software named "Qt". Qt is a cross-platform application framework that is widely used for developing application software with a graphical user interface (GUI) (in which cases Qt is classified as a widget toolkit), and also used for developing non-GUI programs such as command-line tools and consoles for servers. Qt uses standard C++ but makes extensive use of a special code generator (called the *Meta Object Compiler*, or *moc*) together with several macros to enrich the language. Qt can also be used in several other programming languages via language bindings. It runs on the major desktop platforms and some of the mobile platforms. It has extensive internationalization support. Non-GUI features include SQL database access, XML parsing; thread management, network support, and a unified cross-platform API for file handling.

## 5. WORK DIVISION

Code for Steganography : Sudipto Biswas

Code for Encryption :     Sudheer Jhaharia  
                              Surya Kr. Singh (Columnar Transposition cipher)  
                              Suman Naskar (Simple Substitution cipher)  
                              Subhrat Sahoo (Columnar Transposition cipher)

Qt (GUI code) :            TLD Manoj

Explanation of the various parts of our program done by different members is given below. A systematic study of the individual contributions made by different members to the project is done in the following six subsections where each subsection is dedicated to an individual member. All the explanations of the various parts that is given below has been written by individual members.

### 5.1 Sudipto Biswas

I have done the steganography code for the project. Steganography code has been written using the EasyBMP library. It includes four functions namely `getbit()`, `getbitint()`, `Extractmsg()` and `doStegno()`. The function `getbitint()` is used to get the bit values for an int value which is in this code the size of the text file. The other function `getbit()` is used similarly to get the bit values for the characters of the text message to be hidden. Bitwise operators provided by C++ language has been used. The function `doStegno()` has the main code for steganography. It takes two parameters namely the pointers to the file name of the bmp file (in which the message is to be hidden) and the text file (which contains the text message). Again the bitwise operators have been extensively used in this part of the code. Other than that the color values of various pixels have been accessed using the class functions of the EasyBMP class. Then the last function that is the `Extractmsg()` function has been written to destegno the image i.e. get back the message from the stegoimage. It also uses a similar technique to get back the hidden message. To know how much to extract we hide in the first 4\*8 bytes the size of the file which it will use to know where to stop extracting the least significant bits. To know whether the file consists something hidden or not the keyword "\*\*\*\*\*" has been hidden in the next of the 5\*8 bytes of the file. If the `Extractmsg()` on extracting the next 5 bytes from the stegoimage finds it to be the keyword then it will know that there is a message hidden and thus proceed with the extraction or else it will flag a message that no such hidden message exists. Thus these functions together work to give us the steganography code as required.

### 5.2 Sudheer

I have done the modification in encryption & decryption code.

My team mates who were also doing some part of encryption ( transposition) ,they provided me the Basic code of encryption, that was very usefull for me.

Now it reads a \*.txt file named "test.txt" from the user and it gives two output files named "temp.txt" and "output.txt".

The file "temp.txt" contains the encrypted text of the file "test.txt" and the file "output.txt" contains the decrypted text of the file "temp.txt". So finally "output.txt" file is same as "test.txt" file.

I used three functions named " readFile ", " encrypt1 " and "decrypt" in this encryption and decryption code. "readFile" is a simple function that reads text from a \*.txt file and finally returns that text file's data.

We can get that data in a string by calling this function  
i.e. :

```
char* text = readFile("test.txt");
```

Now "test.txt" file's data will be saved in text.

"encrypt1()" is the function for encryption it works like this :

for example:

if we call encrypt1 function in this way"

```
encrypt1("test.txt",key,"temp.txt");
```

suppose text in "test.txt" is " application" and key is "teach"

so there are 5 elements in the key so our given text will be arranged in a 5 cols matrix & it will assign numbers to the key elements in alphabetical order

```
key -->      t e a c h
alp. order --> 4 2 0 1 3
               | a p p l i |
               | c a t i o |
               | n . . . . |
```

now it will create a string like this

```
0- pt
1- li
2- pa
3- io
4- ach
```

string is " ptlipaioach " & it will be saved in " temp.txt " file.

But in the matrix " where , there are ' . ' it will show some arbitrary characters in encrypted part.



NOTE: here whatever i am explaining about encryption to you that is not how compiler does it is the basic understanding method for encryption( transposition) I did encryption with arrays and after I arranged the columns and rows as program needed.

"decrypt" is the function for decryption it works like this :

If we call decrypt function in this way

```
decrypt("temp.txt",key,"Output.txt");
```

It will take input from the "temp.txt" file & works. It will do something like opposite of encryption

basically this function is provided encrypted string and the same key. So it arranges again them in matrix form according to the key and we will get decrypted string from it in another text file named " output.txt ".

## 5.3 TLD Manoj

I've designed the program so that it takes an image from the user and displays it in a imagelabel. takes a text file and displays its contents in a text edit area.

now when the stegno button is clicked, it asks the user for a password to be set on the encryption. then it calls the steganography function and merges the file with the image, and outputs the result image file in a separate imagelabel.

I've discussed this design with my team mates and finalised it.

//programming

I was assigned the gui part of the program. we did the gui part on the ide named qt creator

I used various functions and predefined classes for this project

these include QtGui, QtCore, QWidget, QMainWindow, QTextEdit, QLabel, QScrollArea, QPushButton

and also 3 classes defined by myself, namely, mbox, dialogbox and imageviewer

I found various references from the website "qt.nokia.com", and "stackoverflow.com"

I studied the basic tutorials in qt.nokia.com, and they helped me a lot in understanding the gui layout and implementation.

//testing

I tested the application with different buttons, images and functions and it was working fine. I tried out various layouts and tried using qt designer to augment the look of the project

## 5.4 Surya Kr Singh

I have read about transposition cipher of encryption. Being completely novice to the programming domain I found much difficulty in understanding the basic programming methodologies. So I used much of the time and my project work to build upon my programming skills. At the beginning I started with simple iteration programs, solving a few of them, trying to understand them with the help of my TA and fellow members. So was not able to contribute much to the project code.

## 5.5 Sumon Naskar

I have read about substitution cipher of encryption. Being completely new to the programming domain I found much difficulty in understanding the basic programming methodologies. So I used much of the time and my project work to build upon my programming skills. At the beginning I started with simple iteration programs, solving a few of them, trying to understand them with the help of my TA and fellow members. So was not able to contribute much to the project code.

## 5.6 Subrat Kr Sahoo

I have read about transposition cipher of encryption. Being completely new to the programming domain I found much difficulty in understanding the basic programming methodologies. So I used much of the time and my project work to build upon my programming skills. At the beginning I started with simple iteration programs, solving a few of them, trying to understand them with the help of my TA and fellow members. So was not able to contribute much to the project code.

# 6. SELF-EVALUATION (PEER REVIEWED)

The following table consists of the marks allocated by the team members to themselves after being peer reviewed i.e. the marks that a member has selected to give himself has been approved by all team members.

Roll No	Name	Marks given
110050048	Sudipto Biswas	10
115060018	Suman Naskar	5
110040043	Sudheer Jhajharia	8
110050063	TLD Manoj	9
115060003	Subrat Sahoo	4
115320008	Surya Kr Singh	6

## 7. REFERENCES

<http://en.wikipedia.org/wiki/Steganography>  
<http://en.wikipedia.org/wiki/Encryption>  
[http://en.wikipedia.org/wiki/BMP\\_file\\_format](http://en.wikipedia.org/wiki/BMP_file_format)  
<http://easybmp.sourceforge.net/>  
<http://sector.ynet.sk/qt4-tutorial/my-first-qt-gui-application.html>  
<http://mo.co.za/open/stegoverview.pdf>  
<http://www.infosyssec.com/infosyssec/Steganography/techniques.html>  
[http://en.wikipedia.org/wiki/Qt\\_%28framework%29](http://en.wikipedia.org/wiki/Qt_%28framework%29)  
<http://sector.ynet.sk/qt4-tutorial/my-first-qt-gui-application.html>