Cryptography fundamentals Project (J)

Instructor: MARIMUTHU K Slot: B1

Steganography with Honeywell Encryption

Project Final Review

Prepared by,

17BCI0130	PRASANNA M.V
17BCI0021	HARSHAN S
17BCI0177	THIYAGARAJAN G

ACKNOWLEDGEMENT

The project we worked on is titles as,

“HONEYWELL ENCRYPTION OF TEXT FILE USING IMAGE”.
Firstly, we would like to thank our professor Marimuthu K who gave us his valuable suggestions and ideas when we were in need of them. He encouraged us to work on this project.

We are also grateful to VIT university for giving us the opportunity to work with them and providing us the necessary resources for the project.

We would also thank to all of them who helped us to complete the project. We are immensely grateful to all who involved in this project as without their valuable suggestions it would not have been possible to develop the project within the prescribed time.

Abstract

Security of Confidential data is highly essential in any communications . This motive can be achieved by the use of Encryption with proper keys and algorithms, But for communication of sensitive data there is a requirement of advanced methods of security as to provide less significant exposure of sensitive metadata. This is solved particularly with the help of steganography technique. In this technique we embed the sensitive data in text files and bind along with the image file/picture. This process allows the generalisability of data traffic and avoid sniffing of suspicious traffic which is an additional measure of security for data communication. This method of communicating encrypted message is also termed as honeywell encryption.The implementation of the above mentioned concept is carried out by using scripts to embed the text file into the image but also to avoid the steganalysis and decryption attempts by intruder by reducing the difference in file sizes and the file's outward appearance (picture). This ensures the possibility of sensing abnormality is less and even with successful detection the encryption algorithm is used to strengthen the security enforcement since there is a need of possession of the unique key for decrypting the message. The encrypted message (cipher text) is adaptive in cross platform operating system and it prohibits the intruder to read the content of the image in transit. By proper implementation of the above mentioned concepts we can ensure to provide better security to files that posses sensitive information over existing file encryption algorithms

Keywords: Security, Encryption, Steganography, Communication, honeywell, cipher

1.Introduction

Encryption is one of the important means of securing the internet. Successful encryption is wholly dependent on robust passwords or pass phrases called keys. Data encryption's purpose is to make your files readable and decipherable only to you and those you choose to share that information with. The data in an encrypted file is scrambled into a complex code that cannot be broken within a reasonable amount of time by any computer on earth. The key (or cipher) to unlock this code is a key(passphrase) that is created by whomsoever encrypted the file.

The main aim of our project is to provide a better security to the file that consists of sensitive information. We are working on an algorithm that provides better security than that provided by existing file encrypting algorithms.

2.Problem statement

Currently many cryptography and steganography techniques have come into existence. Encoding of plaintext is achieved using DES, AES, Triple DES, RSA and many other algorithms. Any individual can use his/her one’s own approach as encryption method. Many algorithms such as JSteg, JPHide and JPSeek, OutGuess, F3, F4 and F5 were invented for the purpose of embedding images. These algorithms follow a certain principle to embed and retrieve hidden contents. All the existing approaches have their own disadvantages as they can easily be compromised using steganalysis (A new trend to decipher codes). It means that one way or another, an intruder can figure out the existence of hidden data which results in him/her compromise of sensitive data. Currently, no integrated cryptography and steganography approach in one application exists for image based information security. There are encryption and embedding approaches present that work with plaintext only. The main problem is the attacks that are happening in the cyber world and the threat it poses to the users.

3. Overview

The Encryption scheme being discussed above can be implemented through the following steps

1.Encrypting the text using Caesar cipher with a key.

2.Embedding back inside an image using stenography concept using cmd(modernly developed).

3. At the receivers end, Getting cipher text from the image.

4. Decryption of the cipher-text with the key.

In this project, we added more security that if the attacker decrypt the image and get the cipher text, He doesn’t have the key so even one wrong attempt, using Honey encryption the attacks gets distracted.Hence this provide even more security to the algorithm

4. Proposed Solution

Making the attacker diverted using the honey encryption and also providing the security to both image and the cipher text.
Language used: C++

5. Non Functional Requirements

• **Efficiency** here refers to competency in performance. System performance like time constraint, memory consumption or disk space is important to make sure the efficiency of the application. This requirement can be accomplished by having a quick response time and low process power consumption.

Meaning that the application encrypts the image must be quick with low power usage. The size of the application must be small and should adapt to cross-platform, considering that an intended user will be using various operating system such as Windows, Linux, etc.

• **Security** requirement in this project refer to as the inability of unauthorised user or intruder to read images that are being transmitted between sender and receiver. This is the most crucial requirement for the project.

• The image file sizes should not be significantly affected to prevent doubt for the unauthorised user.

6. Literature Review

The need for secured communication and reduced memory usage are the major issues in recent times. These issues have led to the development of many steganographic algorithms. This paper undertakes the study and implementation of steganographic system with prime concern of secured communication of images along with associated text. An image steganographic model proposed., is modified in such a way that a confidential image is embedded along with the secret text encrypted into single cover image.

Text embedding along with the image is carried out by decomposing the cover image and embedding text in one of the segments.

The one of the algorithm we used is steganographic algorithm.

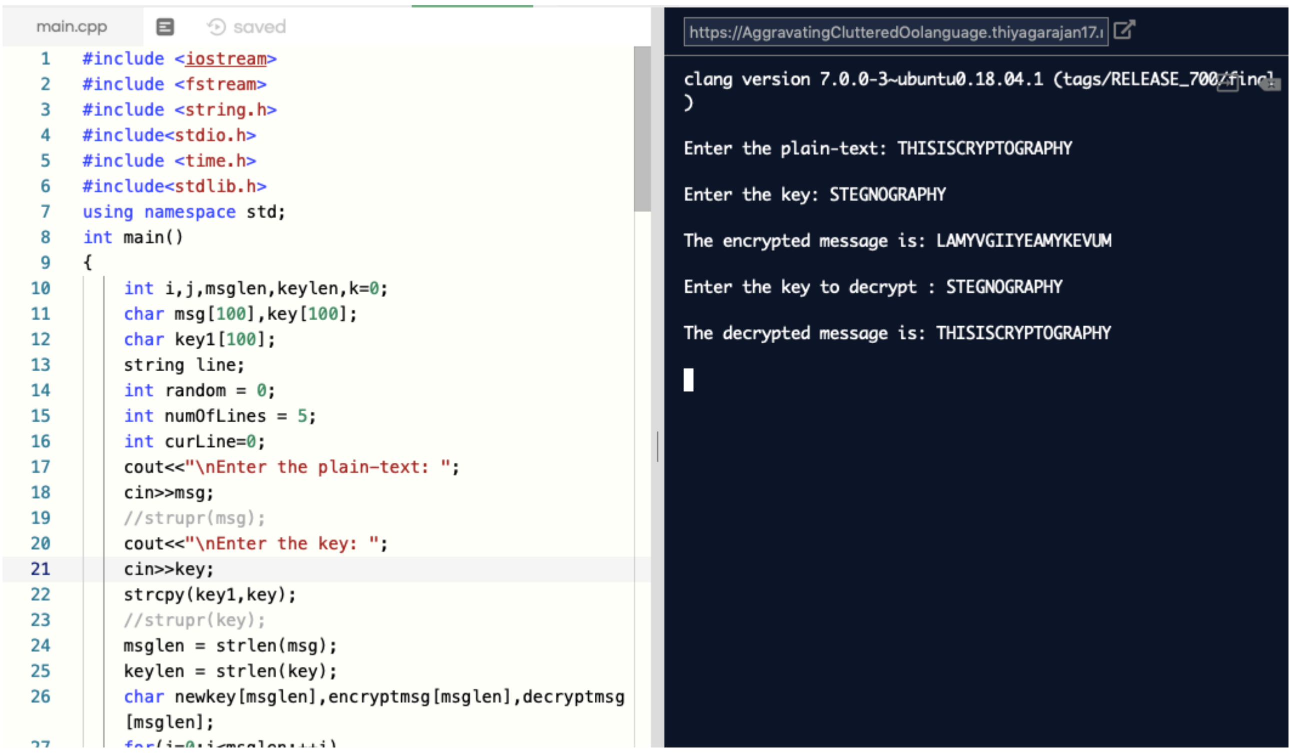
This is the procedure of hiding text in an image such that third party viewers cannot identify that there is text beneath it.

Text embedding along with the image is carried out by decomposing the cover image and embedding text in one of the segments.

Text data is embedded in spatial domain using Least Significant Bit Substitution (LSB) Technique yielding one stego segment.

Since spatial domain techniques are more prone to attacks, text data is encrypted using Advanced Encryption Standard (AES) algorithm for better security. Payload image is embedded into other segment. The other algorithm which we used was “Honey encryption” which are responsible for generating some fake messages to divert the attacker.

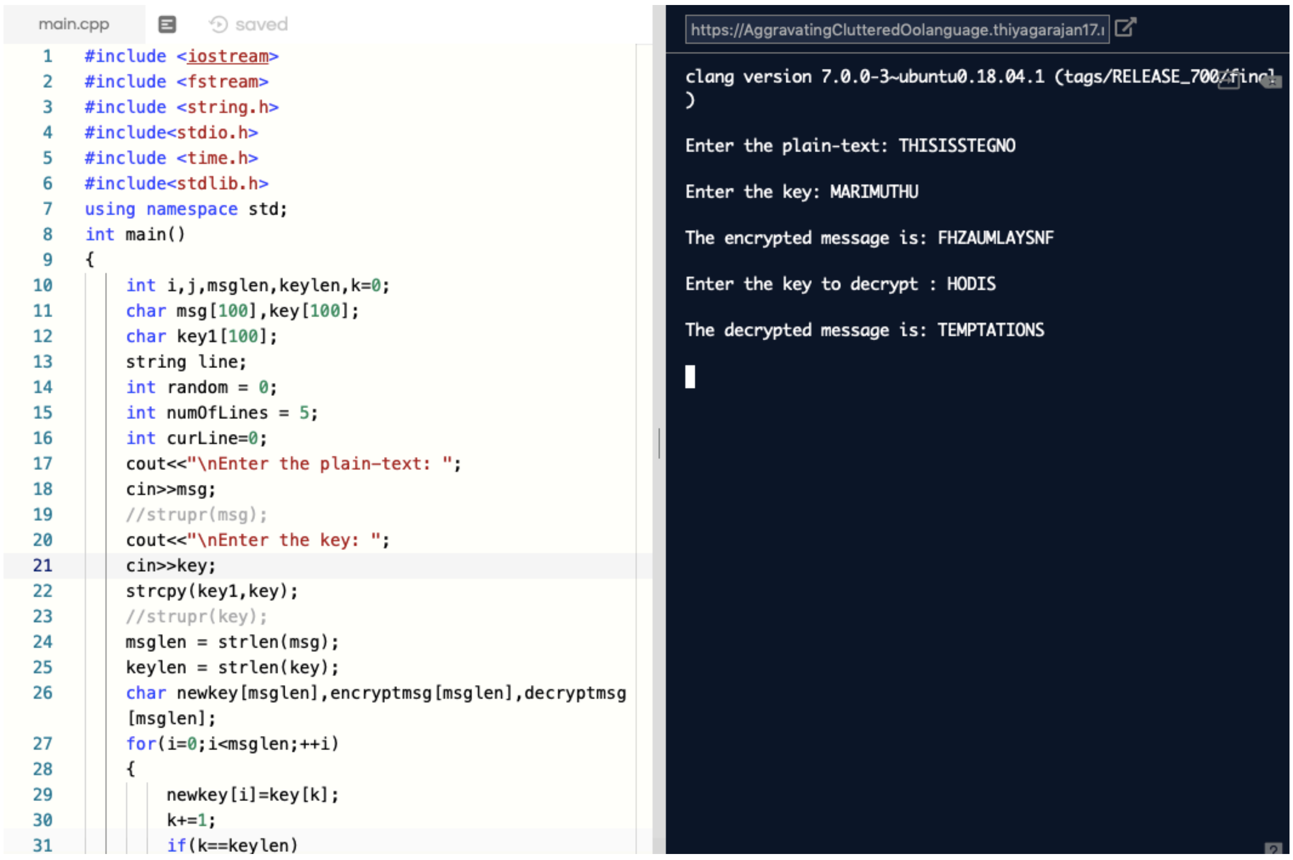
Implementation and Result



```cpp
#include <iostream>
#include <fstream>
#include <string.h>
#include<stdio.h>
#include <time.h>
#include<stdlib.h>
using namespace std;
int main()
{
    int i,j,msglen,keylen,k=0;
    char msg[100],key[100];
    char key1[100];
    string line;
    int random = 0;
    int numOfLines = 5;
    int curLine=0;
    cout<<"\nEnter the plain-text: ";
    cin>>msg;
    //strupr(msg);
    cout<<"\nEnter the key: ";
    cin>>key;
    strcpy(key1,key);
    //strupr(key);
    msglen = strlen(msg);
    keylen = strlen(key);
    char newkey[msglen],encryptmsg[msglen],decryptmsg
    [msglen];
    for(i=0;i<msglen;++i)
```

```
clang version 7.0.0-3~ubuntu0.18.04.1 (tags/RELEASE_700/final
)

Enter the plain-text: THISISCRYPTOGRAPHY

Enter the key: STEGNOGRAPHY

The encrypted message is: LAMYVGIIYEAMYKEVUM

Enter the key to decrypt : STEGNOGRAPHY

The decrypted message is: THISISCRYPTOGRAPHY

▌
```

Honeywell effect for wrong key



```cpp
#include <iostream>
#include <fstream>
#include <string.h>
#include<stdio.h>
#include <time.h>
#include<stdlib.h>
using namespace std;
int main()
{
    int i,j,msglen,keylen,k=0;
    char msg[100],key[100];
    char key1[100];
    string line;
    int random = 0;
    int numOfLines = 5;
    int curLine=0;
    cout<<"\nEnter the plain-text: ";
    cin>>msg;
    //strupr(msg);
    cout<<"\nEnter the key: ";
    cin>>key;
    strcpy(key1,key);
    //strupr(key);
    msglen = strlen(msg);
    keylen = strlen(key);
    char newkey[msglen],encryptmsg[msglen],decryptmsg
    [msglen];
    for(i=0;i<msglen;++i)
    {
        newkey[i]=key[k];
        k+=1;
        if(k==keylen)
```

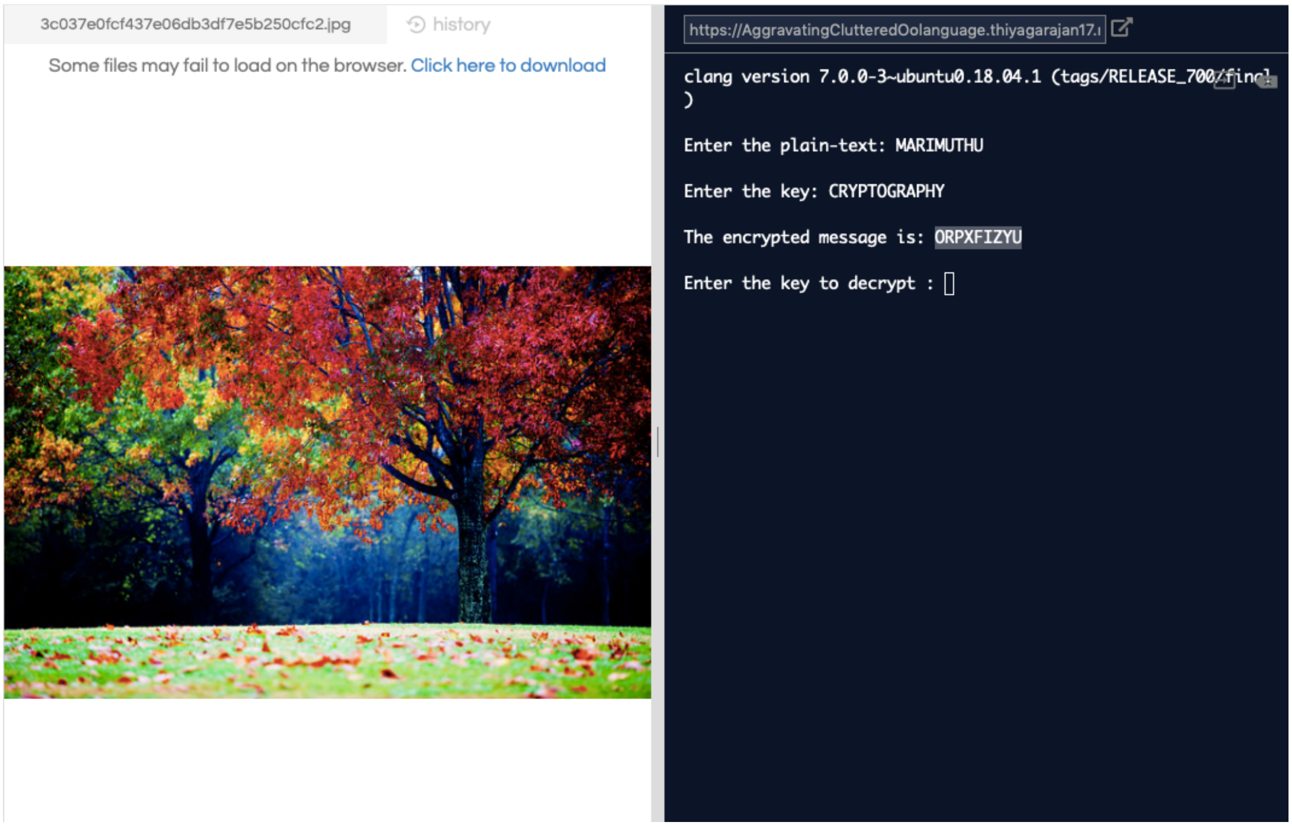
```
clang version 7.0.0-3~ubuntu0.18.04.1 (tags/RELEASE_700/final
)

Enter the plain-text: THISISSTEGNO

Enter the key: MARIMUTHU

The encrypted message is: FHZAUMLAYSNF

Enter the key to decrypt : HODIS

The decrypted message is: TEMPTATIONS

▌
```

We could see the wrong key yields wrong decryption moreover diverts the analyser through providing random meaningful words through which the conventional steganalysis can fail.

Now lets embed the encrypted message into the image . In cmd approach this can be done by the syntax,

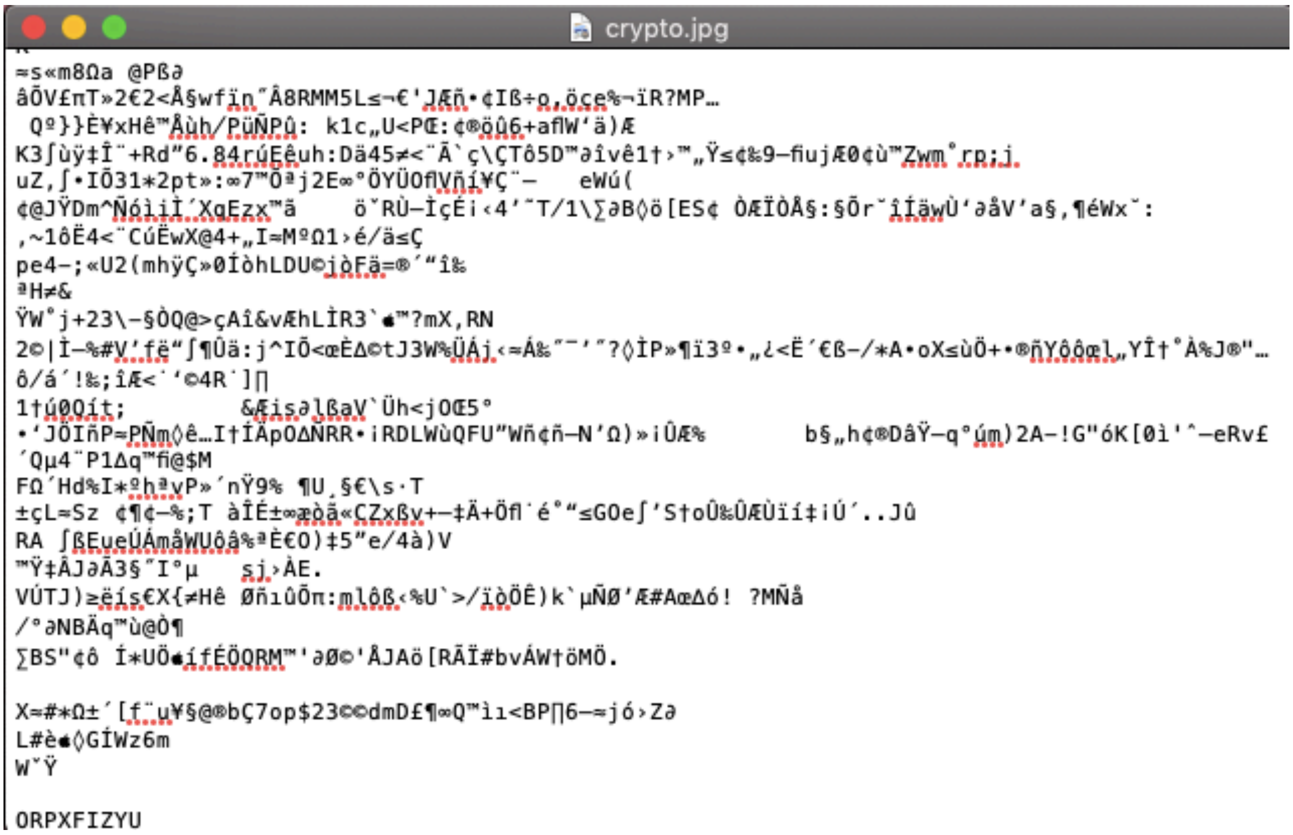
copy /b Name-of-initial-image.jpg + Name-of-file-containing-text-you-want-to-hide.txt Resulting-image-name.jpg

Let's copy the msg to the end of the image file.


3c037e0fcf437e06db3df7e5b250cfc2.jpg
history
Some files may fail to load on the browser. Click here to download
https://AggravatingClutteredOolanguage.thiyagarajan17.

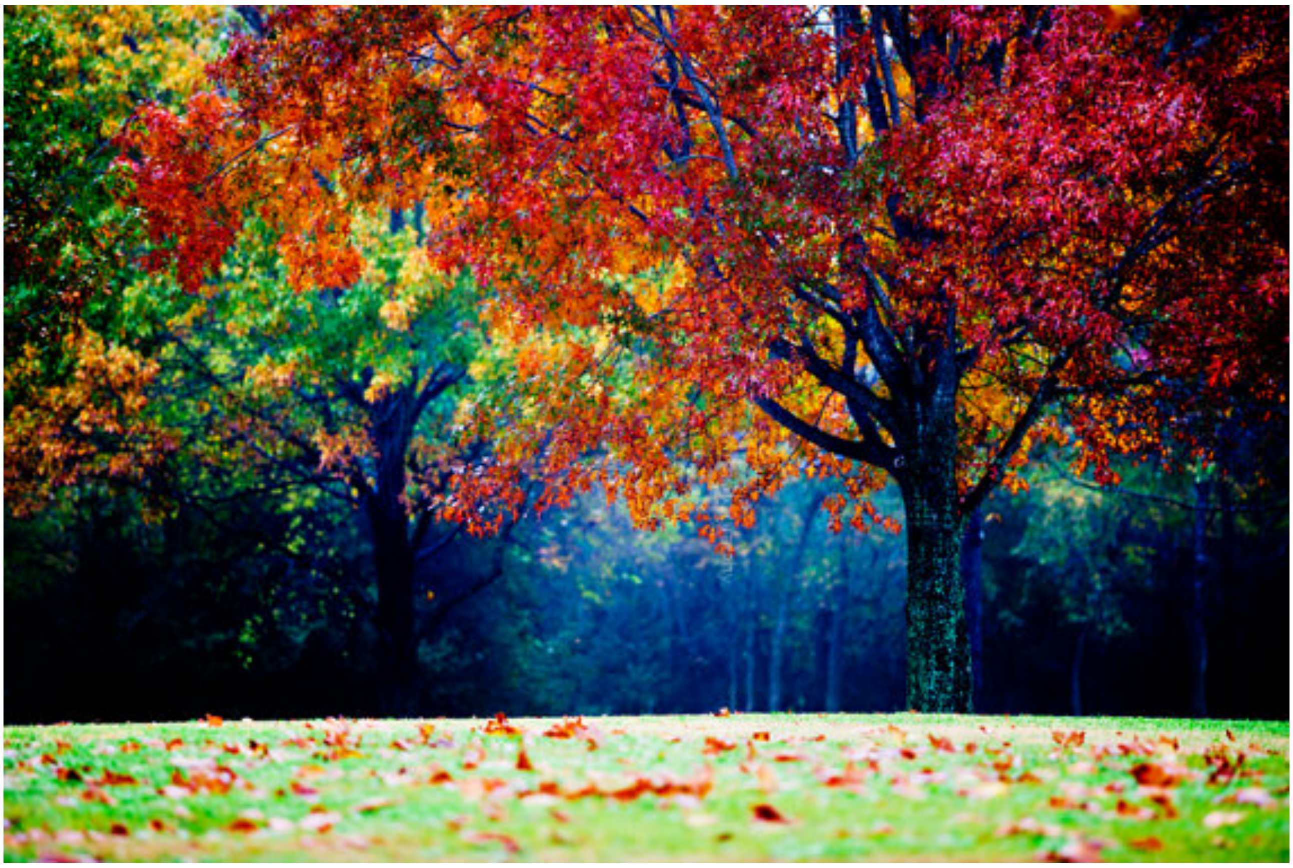
```
clang version 7.0.0-3~ubuntu0.18.04.1 (tags/RELEASE_700/final)
)

Enter the plain-text: MARIMUTHU

Enter the key: CRYPTOGRAPHY

The encrypted message is: ORPXFIZYU

Enter the key to decrypt : 
```

The image file will have the contents as,

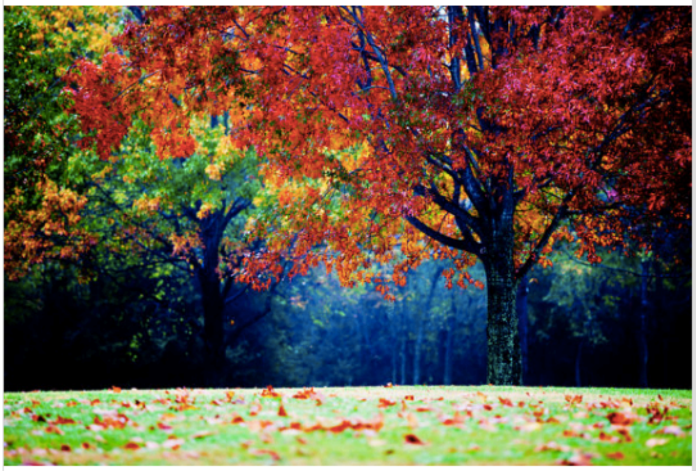


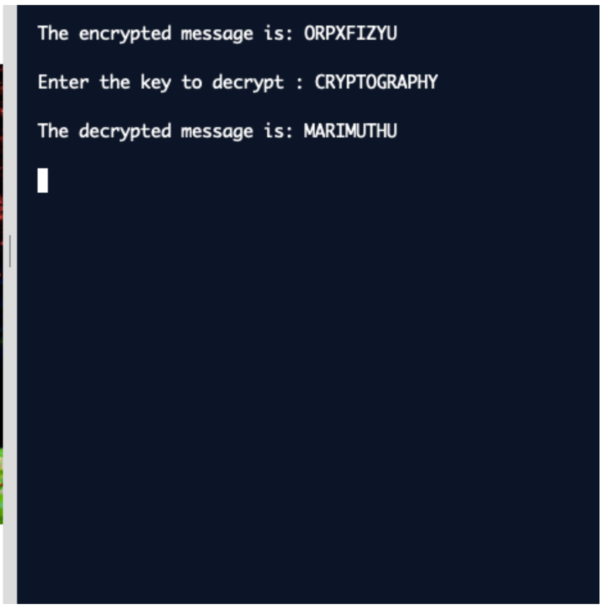
```
≈s«m8Ωa @Pßə
âÕV£πT»2€2<Å§wfïn˝Â8RMM5L≤¬€'JÆñ•¢Iß÷o,öçe%¬ïR?MP…
 Qº}}È¥xHê™Åùh/PüÑPû: k1c„U<PŒ:¢®öû6+aflW‘ä)Æ
K3∫ùÿ‡Î¨+Rd”6.84rúÉêuh:Dä45≠<¨Ā`ç\ÇTô5D™əîvê1†›™„Ÿ≤¢‰9–fiujÆ0¢ù™Zwm˚rp;j
uZ,∫•IÕ31∗2pt»:∞7™Õªj2E∞°ÖYÜOflVñí¥Ç¨–    eWú(
¢@JŸDm^ÑóìiÌ´XqEzx™ã     ö˘RÙ–ÌçÊi‹4’˜T/1\∑əB◊ö[ES¢ ÒÆÏÒÅ§:§Õr˘îÍäwÙ‘əåV’a§,¶éWx˘:
,~1ôË4<¨CúËwX@4+„I≈MºΩ1›é/ä≤Ç
pe4–;«U2(mhÿÇ»0ÍòhLDU©jòFä=®´“î‰
ªH≠&
ŸW˚j+23\–§ÒQ@>çAî&vÆhLÌR3`™?mX,RN
2©|Ì–%#V’fë“∫¶Ûä:j^IÕ<œÈ∆©tJ3W%ÜÁj‹≈Á‰˝¯‘˝?◊ÌP»¶ï3º•„¿<Ë´€ß–/∗A•oX≤ùÖ+•®ñYôôœl„YÎ†˚À%J®"…
ô/á´!‰;îÆ<˙‘©4R˙]∏
1†ú0Qít;           &ÆisəlßaV`Üh<jOŒ5°
•‘JÖIñP≈PÑm◊ê…I†ÍÄpO∆ÑRR•¡RDLWùQFU”Wñ¢ñ–N’Ω)»¡ÛÆ%          b§„h¢®DâŸ–q°úm)2A–!G"óK[0ì'ˆ–eRv£
´Qµ4¨P1∆q™fi@$M
FΩ´Hd%I∗ºhªyP»´nŸ9% ¶U¸§€\s·T
±çL≈Sz ¢¶¢–%;T àÎÉ±∞æòã«CZxßy+–‡Ä+Öfl˙é°“≤G0e∫‘S†oÛ‰ÛÆÙïí‡¡Ú´..Jû
RA ∫ßEueÚÁmåWUôâ%ªÈ€O)‡5”e/4à)V
™Ÿ‡ÂJəĀ3§˝I°µ   sj›ÀE.
VÚTJ)≥ëís€X{≠Hê Øñıûõπ:mlôß‹%U`>/ïòÖÊ)k`µÑØ’Æ#Aœ∆ó! ?MÑå
/°əNBÄq™ù@Ò¶
∑BS"¢ô Í∗UÖíf ÉÖQRM™'əØ©'ÅJAö[RĀÏ#bvÁW†öMÖ.

X≈#∗Ω±´[f¨u¥§@®bÇ7op$23©©dmD£¶∞Q™ìı<BP∏6–≈jó›Zə
L#è◊GÍWz6m
W˘Ÿ

ORPXFIZYU
```

The image carrying the secret message looks exactly the same as the previous one with slight increase in file size



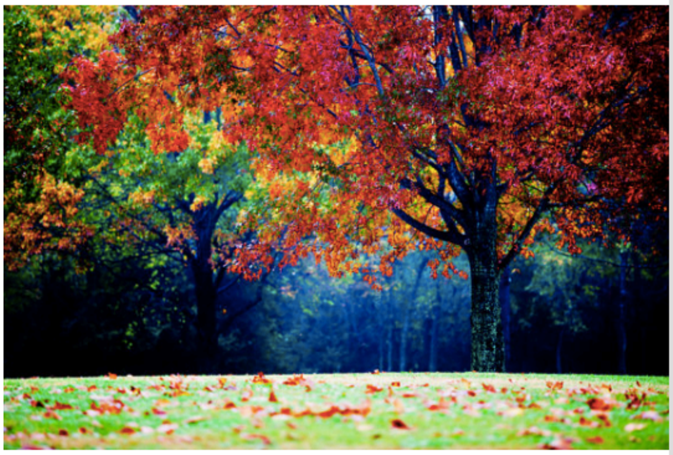
Now when the message is decrypted with proper key we get the proper message

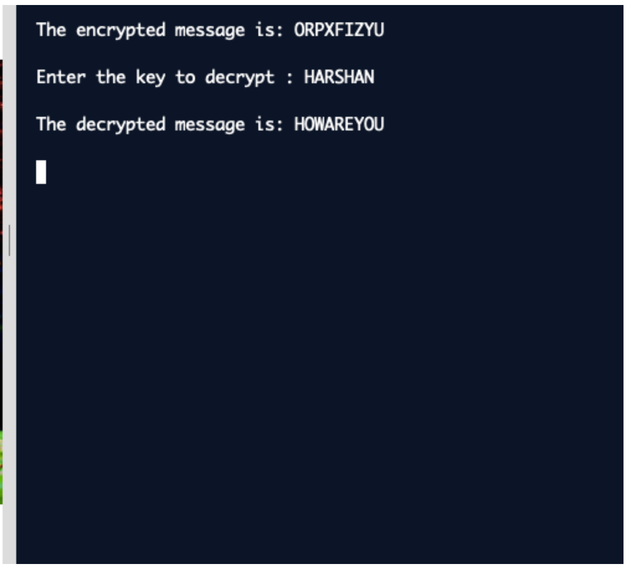




```
The encrypted message is: ORPXFIZYU

Enter the key to decrypt : CRYPTOGRAPHY

The decrypted message is: MARIMUTHU
```

If we enter a wrong key for the input we (attacker) gets diverted by random other texts.





```
The encrypted message is: ORPXFIZYU

Enter the key to decrypt : HARSHAN

The decrypted message is: HOWAREYOU
```

Wrong key supplies "HOWAREYOU" as a message which is diverts the attacker from steganalysis thus providing higher security through honeywell inclusion in stegnography.

—————————————————————————————

CODE:

```
#include <iostream>
#include <fstream>
#include <string.h>
#include<stdio.h>
#include <time.h>
#include<stdlib.h>
using namespace std;
int main()
{
   int i,j,msglen,keylen,k=0;
   char msg[100],key[100];
   char key1[100];
   string line;
```

```
    int random = 0;
    int numOfLines = 5;
    int curLine=0;
    cout<<"\nEnter the plain-text: ";
    cin>>msg;
    //strupr(msg);
    for(int i=0;msg[i]!='\0';i++)
                                                                                                    {

msg[i]=toupper(msg[i]);
                                                                                                    }
    cout<<"\nEnter the key: ";
    cin>>key;
    strcpy(key1,key);
    //strupr(key);
    for(int i=0;key[i]!='\0';i++)
                                                                                                    {

key[i]=toupper(key[i]);
                                                                                                    }
    msglen = strlen(msg);
    keylen = strlen(key);
    char newkey[msglen],encryptmsg[msglen],decryptmsg[msglen];
    for(i=0;i<msglen;++i)
    {
        newkey[i]=key[k];
        k+=1;
        if(k==keylen)
        {
            k=0;
        }
    }
    for(i=0;i<msglen;i++)
          {
             if(msg[i]==' ')
              {
                msg[i]='@';
              }
          }
          //PROVIDE YOUR ENCRYPTION SCHEME
    for(i=0;i<msglen;i++)
    {
        if(msg[i]!='@')
        {
        encryptmsg[i]=((msg[i]+newkey[i])%26)+65;
        }
        else
        {
            encryptmsg[i]=msg[i];
        }
    }
    printf("\t\nThe encrypted message is: ");
```

```
   for(i=0;i<msglen;i++)
   {
      if(encryptmsg[i]!='@')
      {
      cout<<encryptmsg[i];
      }
      else
      {
         cout<<" ";
      }
   }
   cout<<"\n\n\n";

//decrypt
   char chk[20];
   cout<<"\t\nEnter the key to decrypt : ";
   cin>>chk;
   if(strcmp(chk,key1)==0)
   {
      for(i=0;i<msglen;i++)
      {
         if(encryptmsg[i]!='@')
         {
            decryptmsg[i]=(((encryptmsg[i]-newkey[i])+26)%26)+65;
         }
         else
         {
            decryptmsg[i]=encryptmsg[i];
         }
      }
      decryptmsg[i]='\0';
      cout<<"\nThe decrypted message is: ";
      for(i=0;i<msglen;i++)
      {
         if(decryptmsg[i]!='@')
         {
            cout<<decryptmsg[i];
         }
         else
         {
            printf(" ");
         }
      }
   }
   else
   {
      int fklen = 0;
      int n=0;
      ifstream File("words.txt");
      srand(time(0));
      while(random == 0)
      random = rand() % numOfLines;
```

```
        while(getline(File, line))
        {
            ++curLine;
            fklen=line.length();
            if((curLine == random)&&(msglen==fklen))
                cout << line;
            else{
                for(int j=0;j<15;j++){
                    curLine++;
                    if(msglen==fklen)
                    {
                        cout<<"\nThe decrypted message is: "<<line;
                        goto disp;

                    }
                }
            }

        }
    }
    disp:
        cout<<endl<<endl;

    return 0;
}
```

Conclusion:

Strengthening of existing security of information is achieved through combining best encryption scheme with honeywell scheme along with advanced stenography (The text embedded inside the image is hard to recognise as boththth the images with cipherpher text and the image without look alike) methods.

References

- https://stackoverflow.com/questions/16643495/hiding-message-injpg-image
- https://www.codeproject.com/Articles/4336/File-Encryption-andEncrypted-text-embedding-in-an
- https://ieeexplore.ieee.org/document/7019666/?part=1
- https://www.techopedia.com/definition/4131/steganography
- https://en.wikipedia.org/wiki/Steganography