A

Project Report

on

CRYPTOGRAPHY BASED IMAGE/VIDEO STEGANOGRAPHY FOR MORE SECURE DATA HIDING

By

Abhishek Kumar (1784110003)

Akshay Saroj (1784110010)

Rahul Kumar (1784110038)

Sarthak Katiyar (1784110046)

Under the supervision of

Dr. Mainejar Yadav

(Assistant Professor, CSED)

Submitted to the department of Computer Science and Engineering for the partial fulfillment of the requirements for award of Bachelor of Technology

in

COMPUTER SCIENCE AND ENGINEERING

RAJKIYA ENGINEERING COLLEGE, SONBHADRA Dr. A.P.J. Abdul Kalam Technical University, Lucknow, Uttar Pradesh, India, 2020-2021

TABLE OF CONTENTS

Certificate

This is to certify that the Project report entitled "**CRYPTOGRAPHY BASED IMAGE/VIDEO STEGANOGRAPHY FOR MORE SECURE DATA HIDING**" is a record of the work done by the following students:

Student name	**Roll No.**
Abhishek Kumar	1784110003
Akshay Saroj	1784110010
Rahul Kumar	1784110038
Sarthak Katiyar	1784110047

This work is done under our supervision and guidance during the academic year of **2020-21**. This report is submitted to the **Rajkiya Engineering College, Sonbhadra** for partial fulfillment for the degree of **B.TECH. (Computer Science and Engineering) of Dr. A.P.J. Abdul Kalam Technical University,** Lucknow, Uttar Pradesh, India**.**

Sign of Guide:

Dr. Mainejar Yadav

(Assistant Professor,CSED)

ACKNOWLEDGEMENT

Before getting into the thickest of things, we would like to thank the personalities who were part of my project in numerous ways, those who gave me outstanding support from birth of the project.

First and foremost, we would like to thank our guides**, Dr. Mainejar Yadav, Department of Computer Science & Engineering, Rajkiya Engineering College, Sonbhadra** for having suggested the topic of our project and for their constant support and guidance, without which we would not have been able to attempt this project.

We are very obliged to our **beloved Dr. Amod Tiwari, Head of Department of Computer Science & Engineering,** and to our respected **Director Prof. Geetam Singh Tomar, REC Sonbhadra** for permitting us to utilize all the necessary facilities of the institution.

We are highly indebted **to Er. Ashish Ranjan Mishra, Department of Computer Science & Engineering, Rajkiya Engineering College (Sonbhadra)** for support during the tenure of the project.

We hereby wish to express our deep sense of gratitude **to Dr. Anurag Sewak, Department of Computer Science and Engineering, Rajkiya Engineering College (Sonbhadra)** for the esteemed guidance, moral support and invaluable advice provided by him for the success of the project.

We are also thankful to all the staff members of Computer Science and Engineering Department who have co-operate in making our project a success. We would like to thank all our parents and friends who extended their help, encouragement and moral support either directly or indirectly in our project work.

ABSTRACT

Video Steganography is the art and science of encrypting hidden messages within seemingly innocent videos, in such a way that no one other than the intended sender and receiver sees the existence of a hidden message. Steganography uses duplicate parts of video files to embed private messages. Although many different methods of steganography are being discovered and practiced, a good solution has not yet been reached.

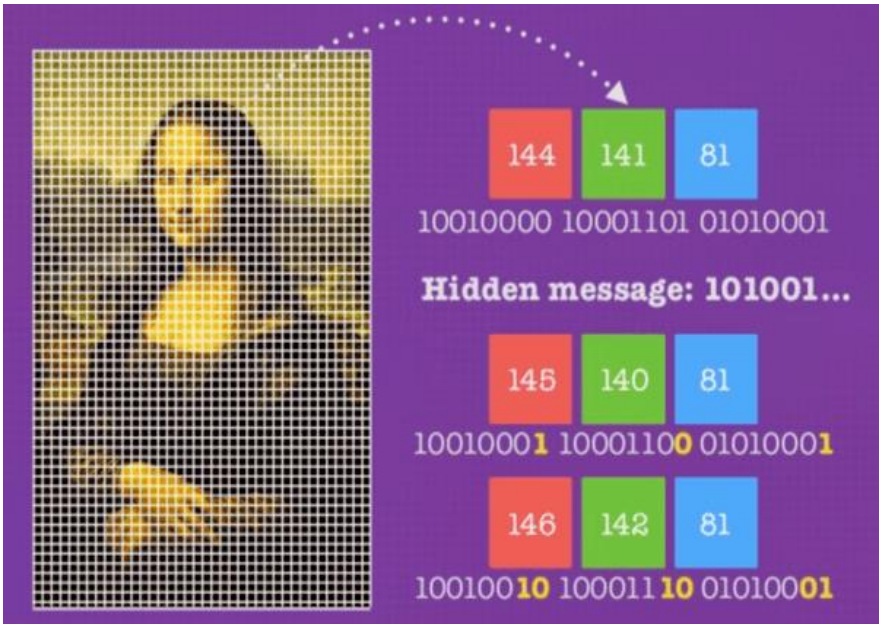
Many new video steganography techniques have been proposed. In this article, some of the most popular methods have been discussed. Most steganography techniques are performed on images, video, text, audio. Steganogrpahic video-based strategies are broadly divided into domain and local domain. In a frequency domain, data is converted into frequency components using FFT, DCT or DWT and the data is added to one or all of the converted coefficients. Embedding can be a low level or a block level. Additionally, in the local domain fragments of data can be inserted in large pixels for LSB video positions. The advantage of the method is that the amount of data (payload) that can be embedded exceeds the LSB strategy. The Steganographic method should not be readily available by an unauthorized person. If a secret message is received with random guesses, the existing steganographic process is considered invalid. Similar to cryptography, steganography may suffer from dangerous attacks.

1. INTRODUCTION

What is steganography?

The term “stegano” is Greek for “hidden” or “covered,” and the “graph” is Greek for “to write.” Put these words together, and we’ve got something close to “hidden writing,” or “secret writing.”

Steganography is the practice of hiding a secret message inside of (or even on top of) something that is not secret.



Steganography in its simplest form:

Performing the invisible ink with lemon juice demonstration is fairly straightforward. Simply dip the cotton swab into the lemon juice and write your message on a piece of paper. The lemon juice “ink” will dry relatively clear and

“invisible.” Once it dries, you can apply heat to the paper to make the message appear and turn brown.

How Steganography Is Used Today?

Steganography can be a way which makes it possible to send news and information without being censored and without the fear of the messages being intercepted and traced back to us.

Today, hackers use PowerShell and BASH scripts to automate attacks. So are pen testers. For example, attackers have been embedding actual scripts within macro-enabled Excel and Word documents. Once a victim opens the Excel or Word doc, they activate the embedded, secret script.

Steganography can be used for digital watermarking in which a message (being simply an identifier) is hidden in an image so that its source can be tracked or verified

1.1 Motivation:

Due to the growing popularity of digital media and the use of digital data, including voice, text, image, and video, in almost all organizations and institutions, the use of steganography to verify digital data origins seems to be in high demand. In recent years, there have been concerns about the growing unauthorized use of such information. However, concern is reduced due to the rapid growth of steganography, watermarking, and encryption methods. In terms of the importance of this issue, this proposal focuses mainly on video steganography.

1.2 Existing Approaches:

B. SUNEETHA et. Al has proposed in his work the Cryptography and Steganography program for encrypting data on video by encrypting it with ASCII

code and providing additional layer of security. Cryptography provides privacy and Steganography is intended to provide privacy.

Kousik Dasgupta, J.K. Mandal and Paramartha Dutta have proposed a hash-based LSB-based video steganography system that uses local domain cover files to hide the presence of sensitive data regardless of how it is formatted. The performance analysis of the LSB-based hash method after comparing the LSB process is better.

A. Swathi, Dr. S.A.K Jilani suggested in his paper that the installation of LSB using a polynomial equation was done to encrypt data in specific video frames and in a specific area of the frame with the installation of LSB using a polynomial number. The key is in the form of polynomial calculations with different coefficients. By using this the ability to cover the pieces in the cover image can be increased.

Mritha Ramalingam suggested a highly secure LSB method where a video file is used as a host media to encrypt a private message without affecting the file structure and content of the video file. Because video quality degradation leads to a noticeable change in video that could lead to the failure of Steganography objectives.

Ashawq T. Hashim et al proposed a Hybrid Encryption and Steganography process where two methods of concealment were used, the first being the Least Significant Bit (LSB) and the second being the Haar Wavelet Transform (HWT). This work is based on a combination of steganography and cryptography techniques to increase the level of security and make the system extremely complex to defeat invaders.

R. Shanthakumari and Drs. Malliga in their proposed work said the LSB Matching Revisited algorithm (LSBMR) selects the embedded regions by the size of the

secret message and the difference between two consecutive pixels in the cover image. The LSBMR scheme addresses two issues Lack of Security and low embedding rate

1.3 Objective:

Steganography is a subset of the Anti-Forensic method used to hide confidential information behind a cover. Steganography is a widely used method of concealing information. The cover can be audio, video, photo or text. In the proposed activity video it is used as a cover to hide private information. In cryptography anyone can easily see that there is confidential information behind a mixed word but in Steganography the presence of confidential information is completely hidden behind an innocent object. No one can easily find out that private information is hidden behind a video. When the data is hidden behind the video we see that there is a change in the size of the actual video and the encrypted video.

With the help of existing techniques, develop a new video steganography technique to hide data/message in carrier video.

1.4 System Overview:

Many new video steganography techniques have been proposed. In this article, some of the most popular methods have been discussed. Most steganography techniques are performed on images, video, text, audio (figure1). Steganogrpahic video-based strategies are broadly divided into domain and local domain. In a frequency domain, data is converted into frequency components using FFT, DCT or DWT and the data is added to one or all of the converted coefficients. Embedding can be a low level or a block level. Additionally, in the local domain fragments of data can be inserted in large pixels for LSB video positions. The advantage of the method is that the amount of data (payload) that can be embedded exceeds the LSB strategy. The Steganographic method should not be readily available by an unauthorized person. If a secret message is received with random

guesses, the existing steganographic process is considered invalid. Similar to cryptography, steganography may suffer from dangerous attacks.

Video steganography is an important area of research in various data encryption technologies, which has become a promising tool because not only is the need for security of sensitive text messages becoming increasingly complex but video is also very popular. In this project, in the form of embedded encrypted messages, video steganography is divided into three categories: internal embedding, pre-set embedding and embedding. Input methods are categorized into video compression categories such as internal prediction, movement forecasts, pixel aggregation, converting coefficients. Pre-embedding methods are used in raw video, which can be separated by spacing and converting domains. Rear embedding methods are more focused on streams, which means that the process of embedding and removal of video steganography is all used in compressed streaming. After that we present a performance test of video steganography and popular video steganography of the future including H.265 video steganography, powerful video steganography and renewable video steganography. And issues are finally discussed in this paper.

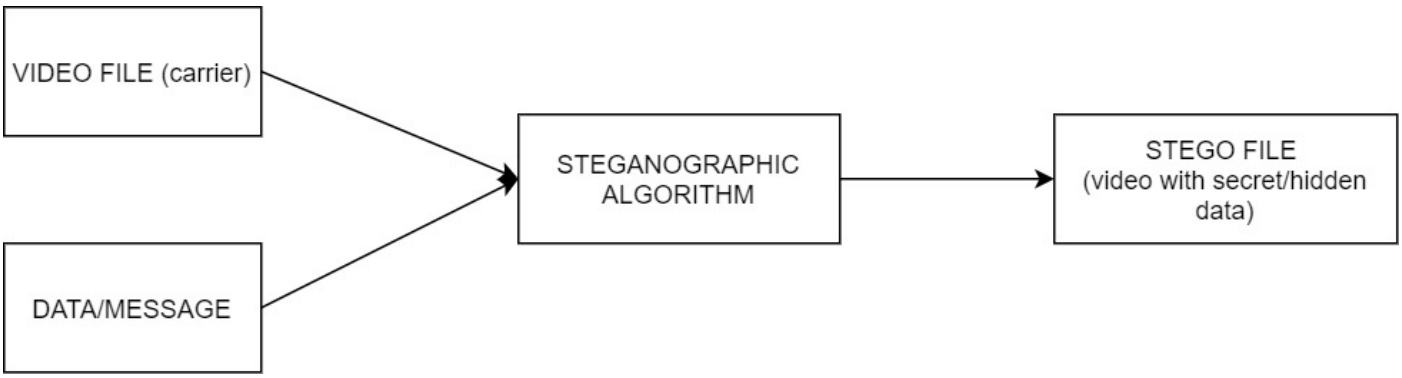


Fig. 1 A generic form of video steganography

2. Literature Review

2.1 Scope:

Video Steganography is the art and science of encrypting hidden messages within seemingly innocent videos, in such a way that no one other than the intended sender and receiver sees the existence of a hidden message. Steganography uses duplicate parts of video files to embed private messages. Although many different methods of steganography are being discovered and practiced, a good solution has not yet been reached.

The Internet makes people's lives easier than ever; they can use it to pay off their debts, buy their goods, exchange important messages between groups in remote areas, and many other things. In addition to protecting that important information, attackers may receive it in various ways. Steganography is one of the ways to protect and hide important information from unauthorized persons and even without suspicion of the existence of information. The Human Visual System (HVS) cannot detect minor changes in media secrecy such as audio, image and video

2.2 Proposed System:

There are two key factors that all successful steganography guidelines need to consider, namely efficiency embedding and earning embedding. First, a highly efficient embedding steganography scheme means good quality stego data and small host (carrier) data to be converted. Any obvious distortions from viewers will increase the chances of suspicion of the attacker and confidential information can be easily obtained with other steganalysis tools. These types of programs are hard to come by for recipients of cooking equipment. The security of the steganography scheme depends directly on the performance of the embedding. Second, the high loading of revenue means that the power of confidential information will be hidden within large hosting data. To be more precise, the two aspects of embedding efficiency and payload embedding have a kind of contradiction. Increasing the efficiency will result in the embedding capacity

having a lower payload. It changes the balance between these two factors mainly rely on the users and the type of steganography methods.

2.3 Problem Definition:

Steganography is a subset of the Anti-Forensic method used to hide confidential information behind a cover. Steganography is a widely used method of concealing information. The cover can be audio, video, photo or text. In the proposed activity video it is used as a cover to hide private information. In cryptography anyone can easily see that there is confidential information behind a mixed word but in Steganography the presence of confidential information is completely hidden behind an innocent object. No one can easily find out that private information is hidden behind a video. When the data is hidden behind the video we see that there is a change in the size of the actual video and the encrypted video.

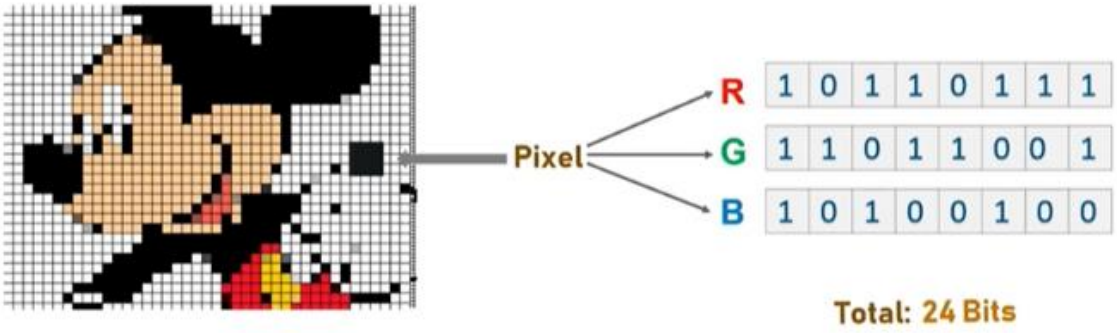
With the help of existing techniques, develop a new video steganography technique with cryptography to hide data/message in carrier video.

3. Least Significant Bit (LSB Method)

Before understanding how can we hide an text inside another, we need to understand what a digital image, pixels are.

Digital image

We can describe a digital image as a finite set of digital values, called pixels. Pixels are the smallest individual element of an image, holding values that represent the brightness of a given color at any specific point. So we can think of an image as a matrix (or a two-dimensional array) of pixels which contains a fixed number of rows and columns.



When using the “digital image” term here, we are referencing to the “raster graphics”, which are basically a dot matrix data structure, representing a grid of pixels, which in turn can be stored in image files with varying formats.

Least Significant Bit (LSB) is a technique in which the last bit of each pixel is modified and replaced with the secret message's data bit.

From the above image it is clear that, if we change MSB it will have a larger impact on the final value but if we change the LSB the impact on the final value is minimal, thus we use least significant bit steganography.

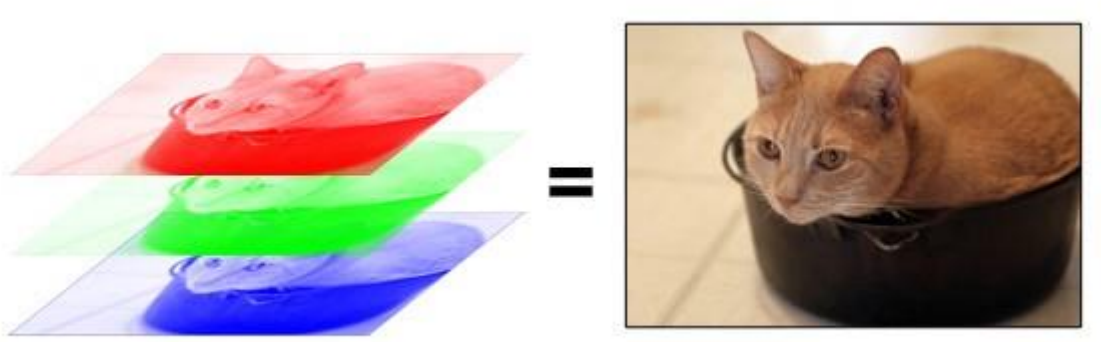
Pixels

Pixels are the smallest individual element of an image. So, each pixel is a sample of an original image. It means, more samples provide more accurate representations of the original. The intensity of each pixel is variable. In color imaging systems, a color is typically represented by three or four component intensities such as red, green, and blue(RGB) or cyan, magenta, yellow, and black(CMYK).

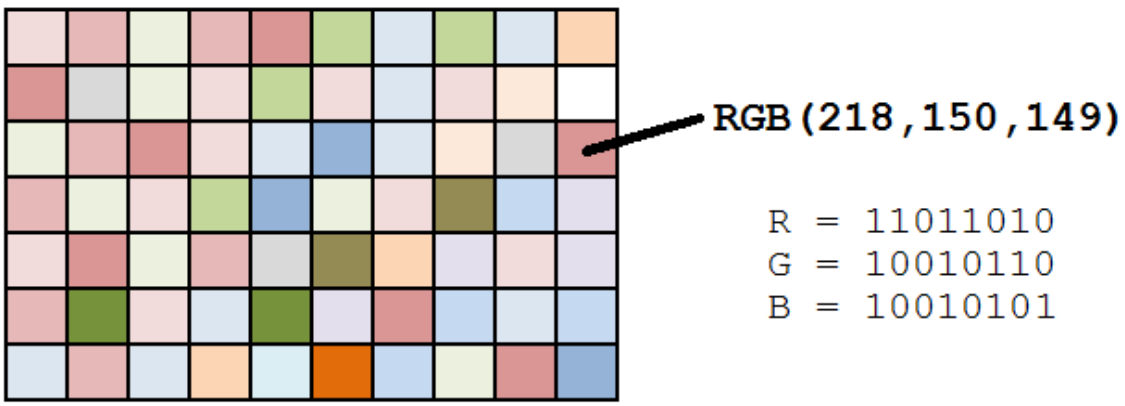
Here, we will work with the RGB color model. As you can imagine, the RGB color model has 3 channels, red, green and blue.

RGB color model

The RGB color model is an additive color model in which red, green and blue light are added together in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three additive primary colors, red, green, and blue. The main purpose of the RGB color model is for the sensing, representation and display of images in electronic systems, such as televisions and computers, though it has also been used in conventional photography.



So, each pixel from the image is composed of 3 values (red, green, blue) which are 8-bit values (the range is 0–255).



As we can see in the image above, for each pixel we have three values, which can be represented in binary code (the computer language).

What is LSB

When working with binary codes, we have more significant bits and less significant bits, as you can see in the image below.

	128	64	32	16	8	4	2	1
8 bit binary digit	1	0	1	1	0	0	0	1
	128 + 32 + 16 + 1 = 177							

Least Significant Bit (LSB) is a technique in which last bit of each pixel is modified and replaced with the data bit. This method only works on Lossless-compression images, which means that the files are stored in a compressed format, but that this compression does not result in the data being lost or modified, PNG, TIFF, and BMP as an example, are lossless-compression image file formats.

As you may already know, an image consists of several pixels, each pixel contains three values (which are Red, Green, Blue), these values range from 0 to 255, in other words, they are 8-bit values. For example, a value of 225 is 11100001 in binary and so on.

The leftmost bit is the most significant bit. If we change the leftmost bit it will have a large impact on the final value. For example, if we change the leftmost bit from 1 to 0 (11111111 to 01111111) it will change the decimal value from 255 to 127.

On the other hand, the rightmost bit is the less significant bit. If we change the rightmost bit it will have less impact on the final value. For example, if we change the leftmost bit from 1 to 0 (11111111 to 11111110) it will change the

decimal value from 255 to 254. Note that the rightmost bit will change only 1 in a range of 256 (it represents less than 1%).

As pixel has three values (RGB), each RGB value is 8-bit (it means we can store 8 binary values) and the rightmost bits are less significant. So, if we change the rightmost bits it will have a small visual impact on the final image. This is the steganography key to hide an image inside another. Change the less significant bits from an image and include the most significant bits from the other image.

How LSB technique works

Each pixel contains three values which are Red, Green, Blue, these values range from **0 to 255**, in other words, they are 8-bit values. Let's take an example of how this technique works, suppose you want to hide the message "**hi**" into a **4x4** image which has the following pixel values:

[(225, 12, 99), (155, 2, 50), (99, 51, 15), (15, 55, 22), (155, 61, 87), (63, 30, 17), (1, 55, 19), (99, 81, 66), (219, 77, 91), (69, 39, 50), (18, 200, 33), (25, 54, 190)]

Using the ASCII Table, we can convert the secret message into decimal values and then into binary: **0110100 0110101.**Now, we iterate over the pixel values one by one, after converting them to binary, we replace each least significant bit with that message bits sequentially (e.g 225 is 11100001, we replace the last bit, the bit in the right (1) with the first data bit (0) and so on).This will only modify the

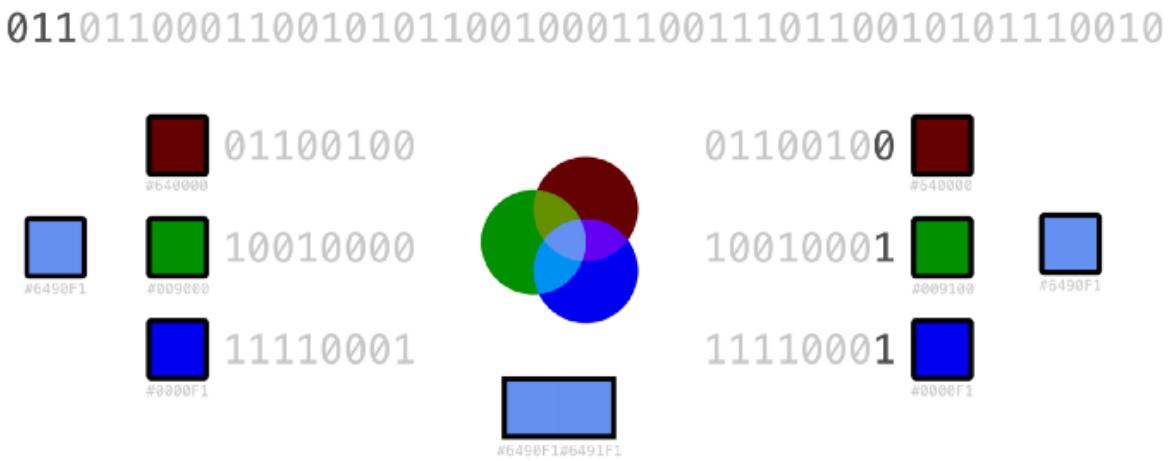
pixel values by +1 or -1 which is not noticeable at all. The resulting pixel values after performing LSBS is as shown below:

[(224, 13, 99), (154, 3, 50), (98, 50, 15), (15, 54, 23), (154, 61, 87), (63, 30, 17), (1, 55, 19), (99, 81, 66), (219, 77, 91), (69, 39, 50), (18, 200, 33), (25, 54, 190)]

How LSB technique works (visually)

Each pixel of the image contains three channels representing the red, green and blue values for that pixel's color. That value can be represented in a single byte and combining them (you can do this with *lighten* blend mode in your image editor) will, of course, return the source color.

Replacing the least significant bit in this case means turning for instance the 01100100 representing the rosewood color to 01100101 but since the first three bits of our data are 011 we only need to replace the green 10010000 with 10010001 . This means that in most cases we wont even have to modify all channels



While the numbers don't lie, the visual difference between #6490F1 and #6491F1 is negligible. Also, we need to keep in mind that when this is done in the real world it would be hosted on an image with a lot more variety of colors which would make it almost impossible for the naked eye to see any difference whatsoever.

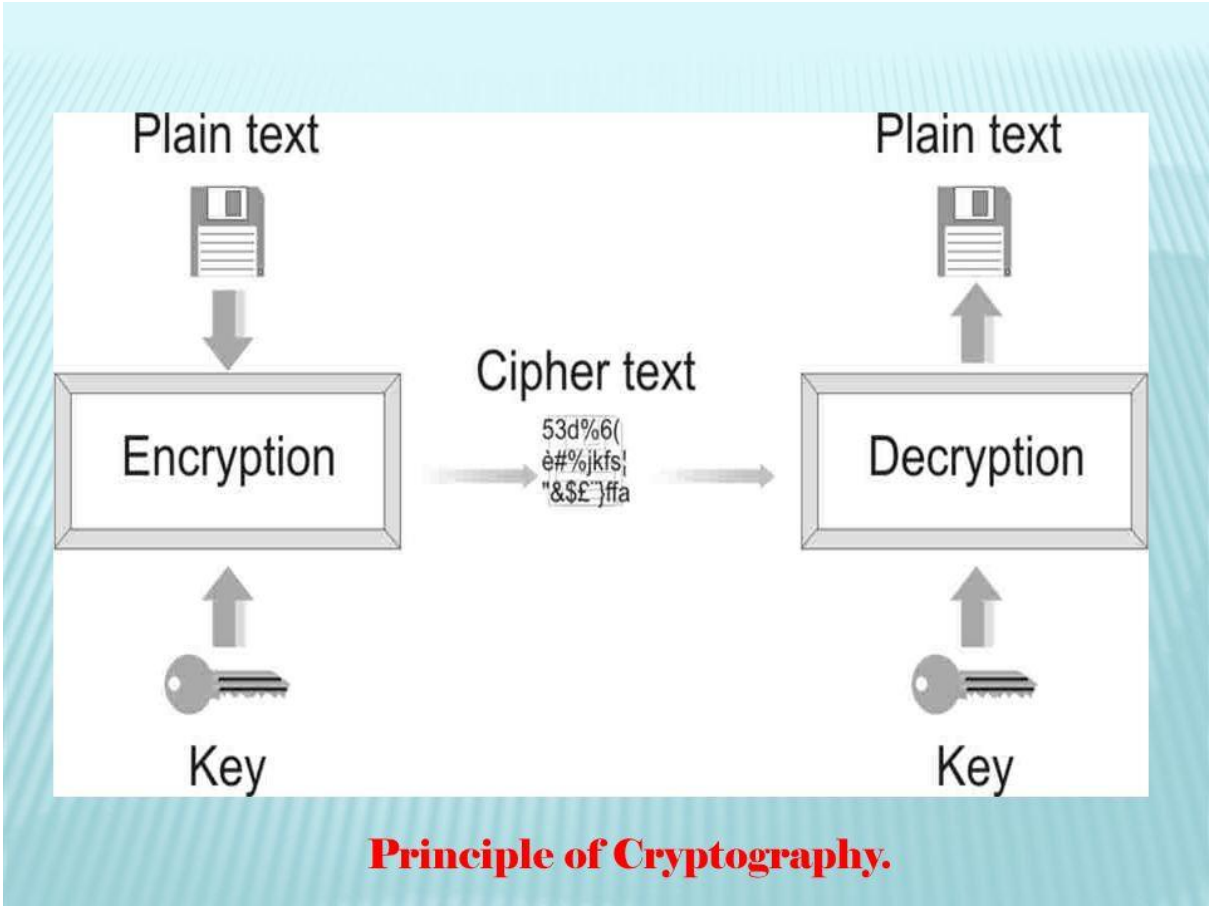
Technique	**Advantages**	**Disadvantages**
Least Signifcant Bit (LSB) Encoding	Hard to detect. Original image is very similar to altered image. Embedded data resembles Gaussian noise.	Message is hard to recover if image is subject to attack such as translation and rotation.
Low Frequency Encoding	Hard to detect as message and fundamental image data share same range.	Significant damage to picture appearance. Message difficult to recover.
Mid Frequency Encoding	Altered picture closely resembles original. Not susceptible to attacks such as rotation and translation.	Relatively easy to detect, as our project has shown.

High Frequency Domain Encoding	None	Image is distorted. Message easily lost if picture subject to compression such as JPEG.

4. CRYPTOGRAPHY

Cryptography is tehnique of securing information and communications through use of codes so that only those person for whom the information intended to be done under it. Therefore prevent unnecessary restrictions access to information. The motto "crypt" means "hidden" and the suffix of the filter "write".

E-Cryptography technique running to protect information and sold bus in the bus ttsttsto These algorithms used as cryptographic key generation, digitally signing, verification to protect data privacy, web browsing on internet and to protect cosmopolitan trasans rosary rosas hotel and prizes



Principle of Cryptography.

Types Of Cryptography:

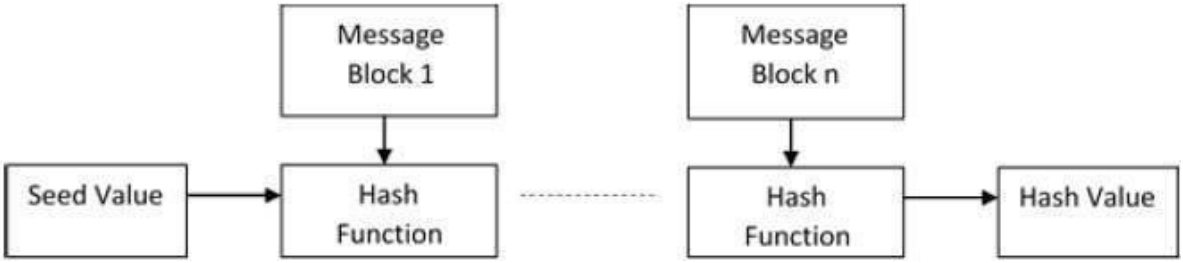
In general there are three types Of cryptography:

Symmetric Key Cryptography:

It is in the encryption program where the sender also recovers and uses a single use of the common key to encrypt and decrypt messages. Symmetric Key Settings are active and painful but problem is the sender and removes the sender when you release the key to the mann. The most advanced symmetric key syringes key system is the Datry Enlightenment System (DES).

Hash Function:

There is a key no usage of any in this algorithm. An open value of limited length is extracted to obtain plain text while it is understandable how it can be accessed if it wants to be rewritten. Many operating programs use hash functions to encrypt passwords.



Asymmetric Key Cryptography:

Under this system a pair of keys is used to encrypt and decrypt information. A public key is used for encryption and a private key is used for decryption. Public key and Private Key are different. Even if the public key is known by everyone the intended receiver can only decode it because he alone knows the private key.

How Encryption Works:

Sensitive screen with limited numbers over padlock over numbers - encryption conceptEncryption isaa way for data-messages or files-to be made unreadable, to ensure it will not be purchased. Encryption uses complex algorithms to scramble data and decrypts the same data using a key provided by the message sender. Encryption ensures that information stays private and confidential, regardless of whether they are made of transit. Anything not included in the data will only detect bytes.

Algoritham:

Also known as a cipher, algorithms are the rules or instructions for the encryption process. The key length, functionality, and features of the encryption system in use determine the effectiveness of the encryption.

Common Encryption Algorithms :

1. **Triple DES:**

Triple DES was designed to replace the original Data Encryption Standard (DES) algorithm, which hackers eventually learned to defeat with relative ease. At one time, Triple DES was the recommended standard and the most widely used symmetric algorithm in the industry.

Triple DES uses three individual keys with 56 bits each. The total key length adds up to 168 bits, but experts would argue that 112-bits in key strength is more accurate. Despite slowly being phased out, Triple DES has, for the most part, been replaced by the Advanced Encryption Standard (AES).

AES:

The Advanced Encryption Standard (AES) is the algorithm trusted as the standard by the U.S. Government and numerous organizations. Although it is highly efficient in 128-bit form, AES also uses keys of 192 and 256 bits for heavy-duty encryption purposes.

AES is largely considered impervious to all attacks, except for brute force, which attempts to decipher messages using all possible combinations in the 128, 192, or 256-bit cipher.

RSA Security:

RSA is a public-key encryption algorithm and the standard for encrypting data sent over the internet. It also happens to be one of the methods used in PGP and GPG programs. Unlike Triple DES, RSA is considered an asymmetric algorithm due to its use of a pair of keys. You've got your public key to encrypt the message and a private key to decrypt it. The result of RSA encryption is a huge batch of mumbo jumbo that takes attackers a lot of time and processing power to break.

Blowfish:

Blowfish is yet another algorithm designed to replace DES. This symmetric cipher splits messages into blocks of 64 bits and encrypts them individually. Blowfish is known for its tremendous speed and overall effectiveness. Meanwhile, vendors have taken full advantage of its free availability in the public domain. It's one of the more flexible encryption methods available.

Twofish:

Computer security expert Bruce Schneier is the mastermind behind Blowfish and its successor Twofish. Keys used in this algorithm may be up to 256 bits in length, and as a symmetric technique, you only need one key.

Twofish is one of the fastest of its kind and ideal for use in hardware and software environments. Like Blowfish, Twofish is freely available to anyone who wants to use it.

Why to use AES 256 in Project:

AES has proved to be very effective and efficient and, given the correct key, adds little to no-noticeable difference in overhead for any process where it is utilized. Essentially, AES is a fast and highly secure form of encryption.

With a 256-bit encryption key, AES is very secure virtually unbreakable.

The Types of Cryptographic Hash Algorithms:

1. **Message Digest (MD):**

MD5 was most popular and widely used hash function for quite some years.

The MD family comprises of hash functions MD2, MD4, MD5 and MD6. It was adopted as Internet Standard RFC 1321. It is a 128-bit hash function.

MD5 digests have been widely used in the software world to provide assurance about integrity of transferred file. For example, file servers often provide a pre-computed MD5 checksum for the files, so that a user can compare the checksum of the downloaded file to it.

In 2004, collisions were found in MD5. An analytical attack was reported to be successful only in an hour by using computer cluster. This collision attack resulted in compromised MD5 and hence it is no longer recommended for use.

2. **Secure Hash Function(SHA):**

Family of SHA comprise of four SHA algorithms; SHA-0, SHA-1, SHA-2, and SHA-3. Though from same family, there are structurally different.

The original version is SHA-0, a 160-bit hash function, was published by the National Institute of Standards and Technology (NIST) in 1993. It had few weaknesses and did not become very popular. Later in 1995, SHA-1 was designed to correct alleged weaknesses of SHA-0.

SHA-1 is the most widely used of the existing SHA hash functions. It is employed in several widely used applications and protocols including Secure Socket Layer (SSL) security.

In 2005, a method was found for uncovering collisions for SHA-1 within practical time frame making long-term employability of SHA-1 doubtful.

SHA-2 family has four further SHA variants, SHA-224, SHA-256, SHA-384, and SHA-512 depending up on number of bits in their hash value. No successful attacks have yet been reported on SHA-2 hash function.

Though SHA-2 is a strong hash function. Though significantly different, its basic design is still follows design of SHA-1. Hence, NIST called for new competitive hash function designs.

In October 2012, the NIST chose the Keccak algorithm as the new SHA-3 standard. Keccak offers many benefits, such as efficient performance and good resistance for attacks.

3. **RIPEMD:**

The RIPEMD is an acronym for RACE Integrity Primitives Evaluation Message Digest. This set of hash functions was designed by open research community and generally known as a family of European hash functions.

The set includes RIPEMD, RIPEMD-128, and RIPEMD-160. There also exist 256, and 320-bit versions of this algorithm.

Original RIPEMD (128 bit) is based upon the design principles used in MD4 and found to provide questionable security. RIPEMD 128-bit version came as a quick fix replacement to overcome vulnerabilities on the original RIPEMD.

RIPEMD-160 is an improved version and the most widely used version in the family. The 256 and 320-bit versions reduce the chance of accidental collision, but do not have higher levels of security as compared to RIPEMD-128 and RIPEMD-160 respectively.

Whirlpool :

This is a 512-bit hash function.

It is derived from the modified version of Advanced Encryption Standard (AES). One of the designer was Vincent Rijmen, a co-creator of the AES.

Three versions of Whirlpool have been released; namely WHIRLPOOL-0, WHIRLPOOL-T, and WHIRLPOOL.

Working of Hash algorithm in project:

It is used to **Message Digest 5 (MD5)** Hash algorithm.

Step 1: Append Padding Bits:

Padding means adding extra bits to the original message. So in MD5 original message is padded such that its length in bits is congruent to 448 modulo 512. Padding is done such that the total bits are 64 less, being a multiple of 512 bits length.

Padding is done even if the length of the original message is already congruent to 448 modulo 512. In padding bits, the only first bit is 1, and the rest of the bits are 0.

Step 2: Append Length:

After padding, 64 bits are inserted at the end, which is used to record the original input length. Modulo 2^64. At this point, the resulting message has a length multiple of 512 bits.

Step 3: Initialize MD buffer:

A four-word buffer (A, B, C, D) is used to compute the values for the message digest. Here A, B, C, D are 32- bit registers.

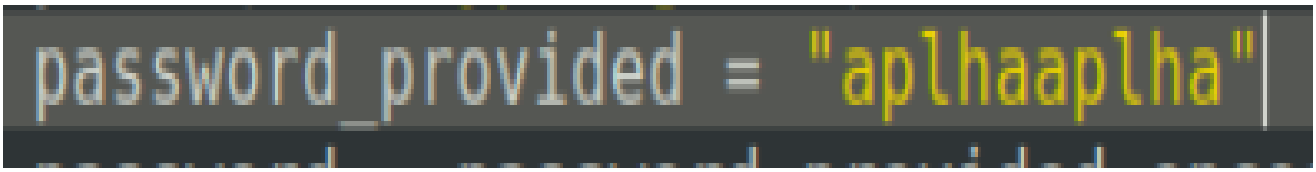
Step 4: Processing message in 16- word block :

MD5 uses the auxiliary functions, which take the input as three 32-bit numbers and produce 32-bit output. These functions use logical operators like OR, XOR, NOR.

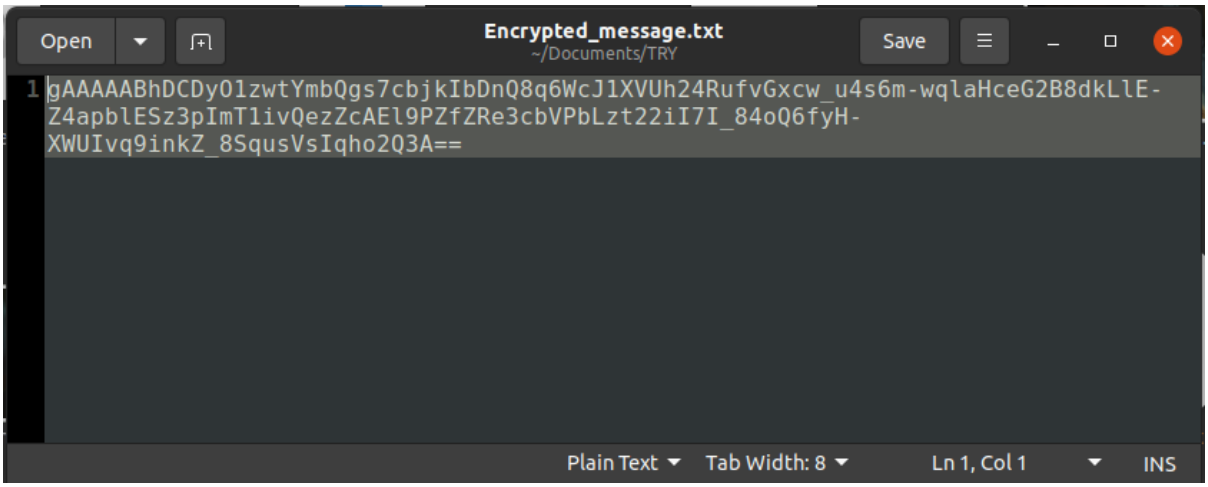
The content of four buffers are mixed with the input using this auxiliary buffer, and 16 rounds are performed using 16 basic operations.

After all, rounds have performed, the buffer A, B, C, D contains the MD5 output starting with lower bit A and ending with higher bit D.

Input:



Output:



5. Encoding

In this part the data (image/video pixel) that is inputted by the user is being encoded with the secret message that is to be hide inside the data pixel(image/video).

This Hiding (embedding) of the data is done with help of following code in which the data (image/video) is divided into its pixel.

Once the division of the pixel is achieved than the secret message that is to be transmitted is break into bit size information.

And, this bit size information of the secret message is placed very carefully in the place of the least significant bit (LSB) in each pixel.

This is how, embedding of the secret message into the image/video data is achieved.

The Code:

```
import cv2

import os

import base64

from stegano import lsb

from subprocess import call, STDOUT

from pathlib import Path

from cryptography.fernet import Fernet

from cryptography.hazmat.backends import default_backend
```

```
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.kdf.pbkdf2 import PBKDF2HMAC



print("Cryptography Based Image/Vide Steganography for more secure data hiding")
print("ENCODING MODULE")
print("1.Hide Message in Image")
print("2.Hide Message in Video")

choice = input()



if choice == '1':
    print("Enter Image Path")
    img_path = input()
    print("Enter Message")
    message = input()
    print("Encoding...")
    secret=lsb.hide(img_path,message)
    secret.save('./encoded_image.png')
    print("Completed.....")

elif choice == '2':
```

```
print("Encrypting...")
password_provided = "katiyar"
password = password_provided.encode()


salt = b'&\xb8\xd9\xa5\x92\x00Pl\x92Y\xf6\xd8#\x92\xcdC'
kdf = PBKDF2HMAC(
    algorithm=hashes.SHA256(),
    length=32,
    salt=salt,
    iterations=100000,
    backend=default_backend()
)

key = base64.urlsafe_b64encode(kdf.derive(password))
print ("Encryption-Key:")
print (key)
file = open('Encryption_key.key','wb')
file.write(key)
file.close()


txt = Path('message.txt').read_text()
txt = txt.replace('\n','')
```

```
encoded = txt.encode()



f = Fernet(key)
encrypted = f.encrypt(encoded)
# print(encrypted)
file = open('Encrypted_message.txt','wb')
file.write(encrypted)
file.close()



print("Cipher-Text:")
print(encrypted)
def split(word):
    return [char for char in word]



 call(["ffmpeg", "-i","video.mp4" , "-q:a", "0", "-map", "a",
"temp_folder/audio.mp3", "-y"],stdout=open(os.devnull, "w"), stderr=STDOUT)



print("Extracting frams from the video...")
vidcap = cv2.VideoCapture("video.mp4")
count=0
```

```
    while True:
        success, image = vidcap.read()
        if not success:
            break
        cv2.imwrite(os.path.join("temp_folder", "{:d}.png".format(count)), image)
        count += 1

    print("Embedding the Encrypted text in frames...")
    txt = Path('Encrypted_message.txt').read_text()
    txt = txt.replace('\n','')
    str = split(txt)
    for i in range(0,len(str)):
        f_name = "{}{}.png".format("temp_folder/",i)
        secret = lsb.hide(f_name,str[i])
        secret.save(f_name)
        print("[INFO] frame {} holds {}".format(f_name,str[i]))

    print("Compiling the video...")
    call(["ffmpeg", "-i", "temp_folder/%d.png" , "-vcodec", "png", "video.mov", "-
y"],stdout=open(os.devnull, "w"), stderr=STDOUT)

    call(["ffmpeg", "-i", "video.mov", "-i", "temp_folder/audio.mp3", "-codec",
"copy","data/enc-" +"final"+".mov", "-y"],stdout=open(os.devnull, "w"),
stderr=STDOUT)
    print("Completed.")
```

```
else:
    print("Invalid Choice!")
```

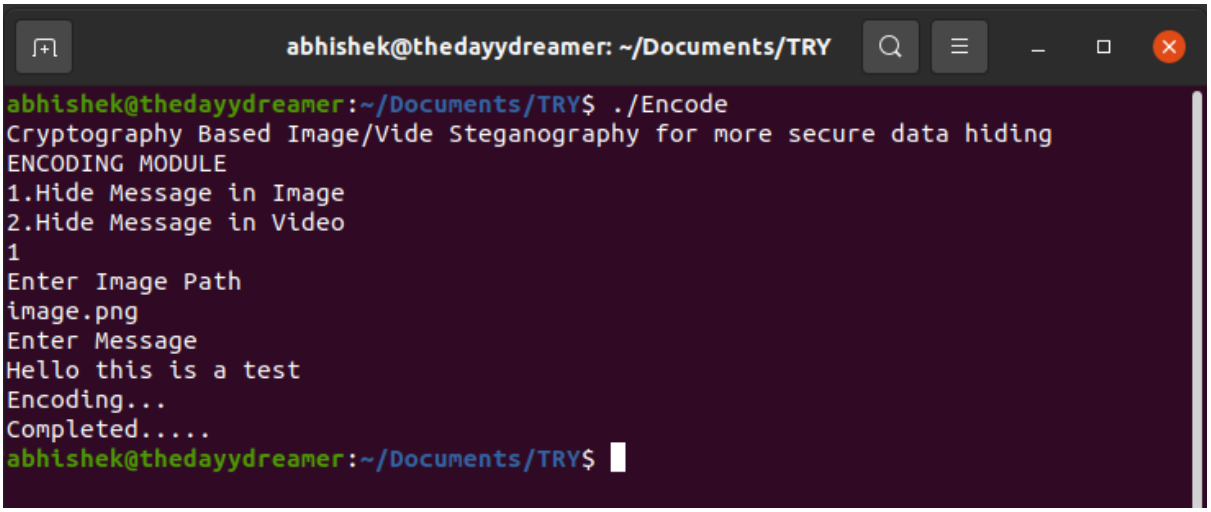


Figure 1 - Data Encoding

When the code is executed, the above screen (figure-1) is obtained.

The User here required to enter the address of the data (image/video data) that is to be used for hiding the secret message.

A secret message (Hello this is test…) in (figure-2) is also provided by the user that is to be embedded (hide) in the data (image/video).

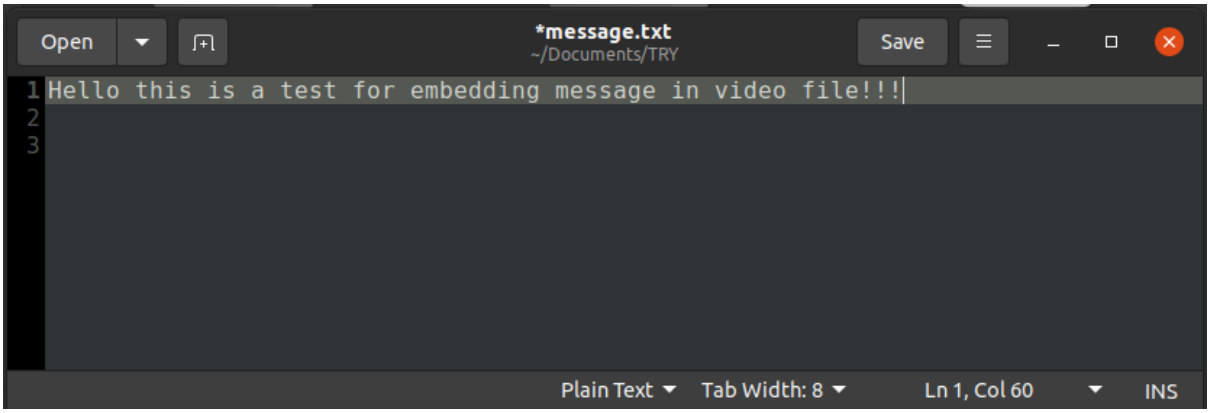


Figure 2 – Secret Message

And when the user the required data and secret message into the system the encoding of the data will takes place.

After the execution of the code a success message appears that encoding completed.

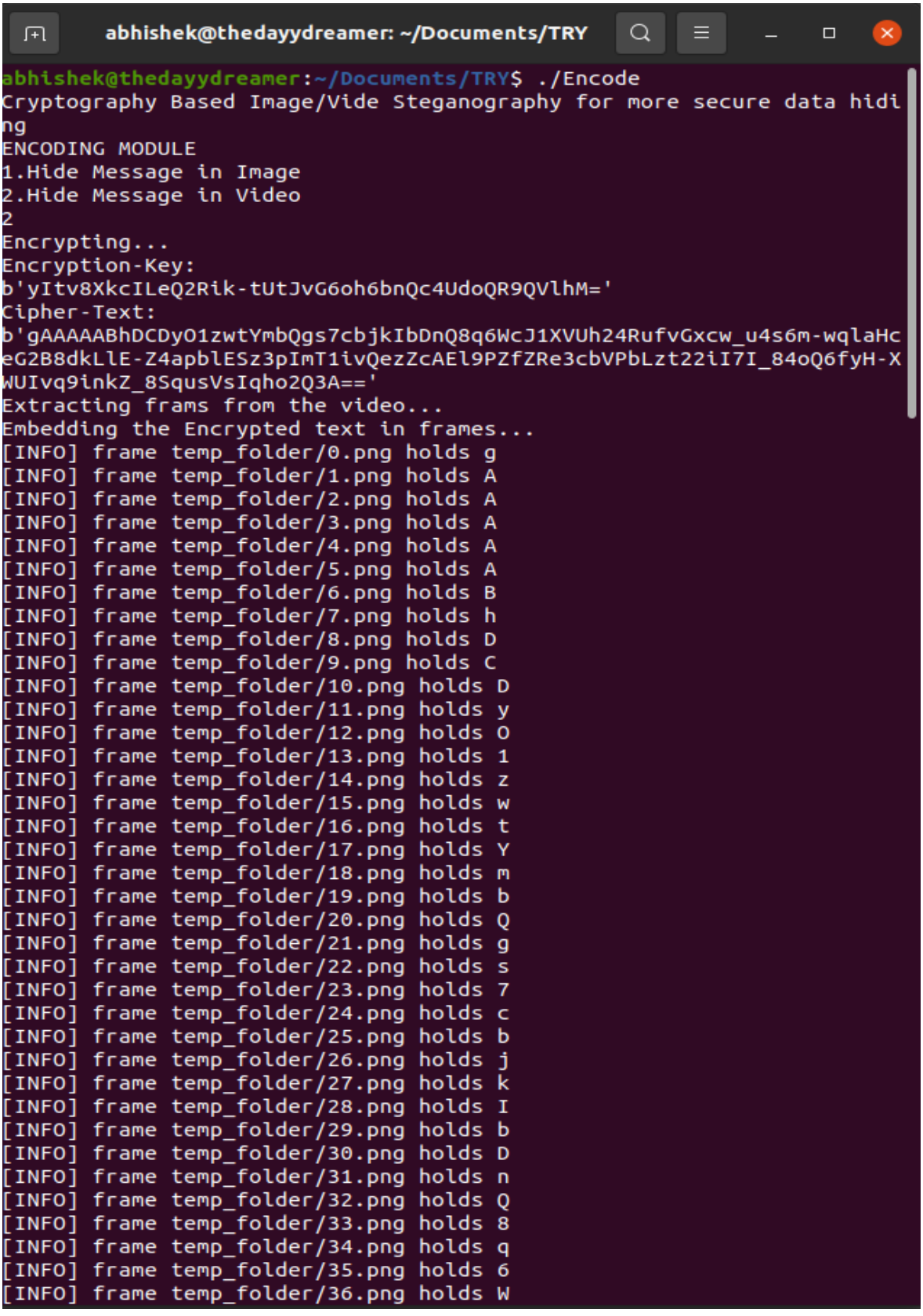


Figure 3 – Encoding data into frames

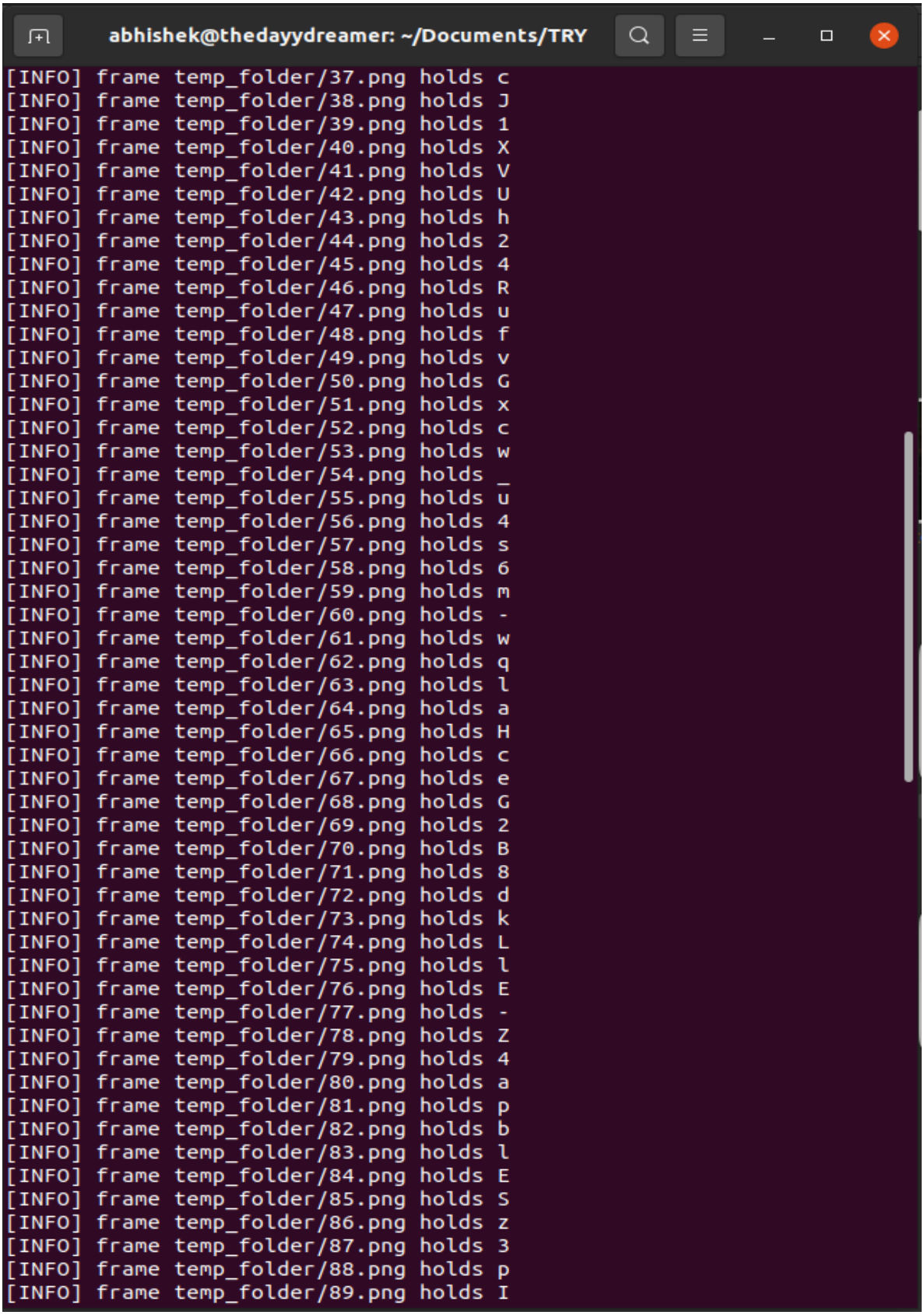


Figure 4 – Encoding data in frames 2

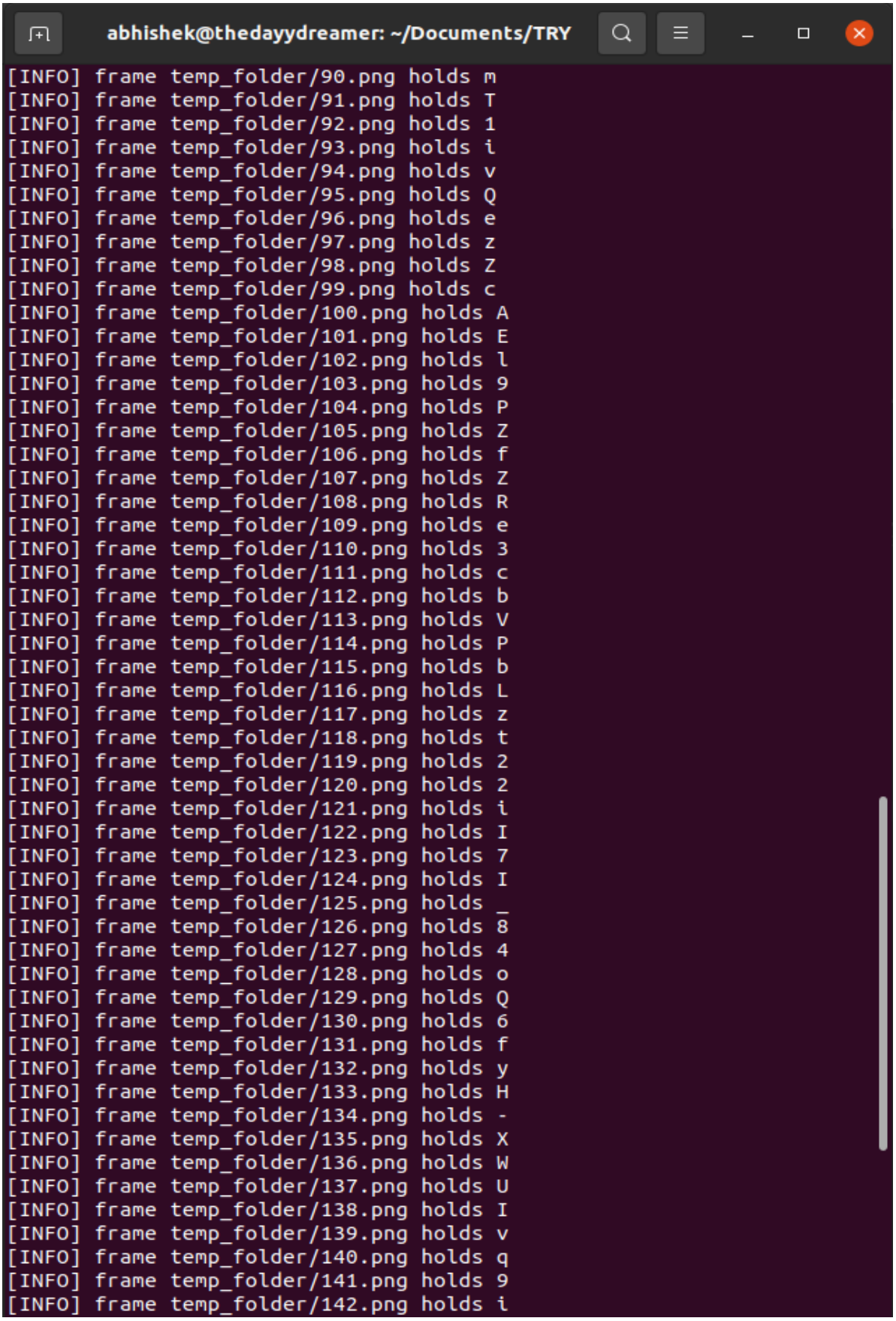


Figure 5 – Encoding data in frames 3

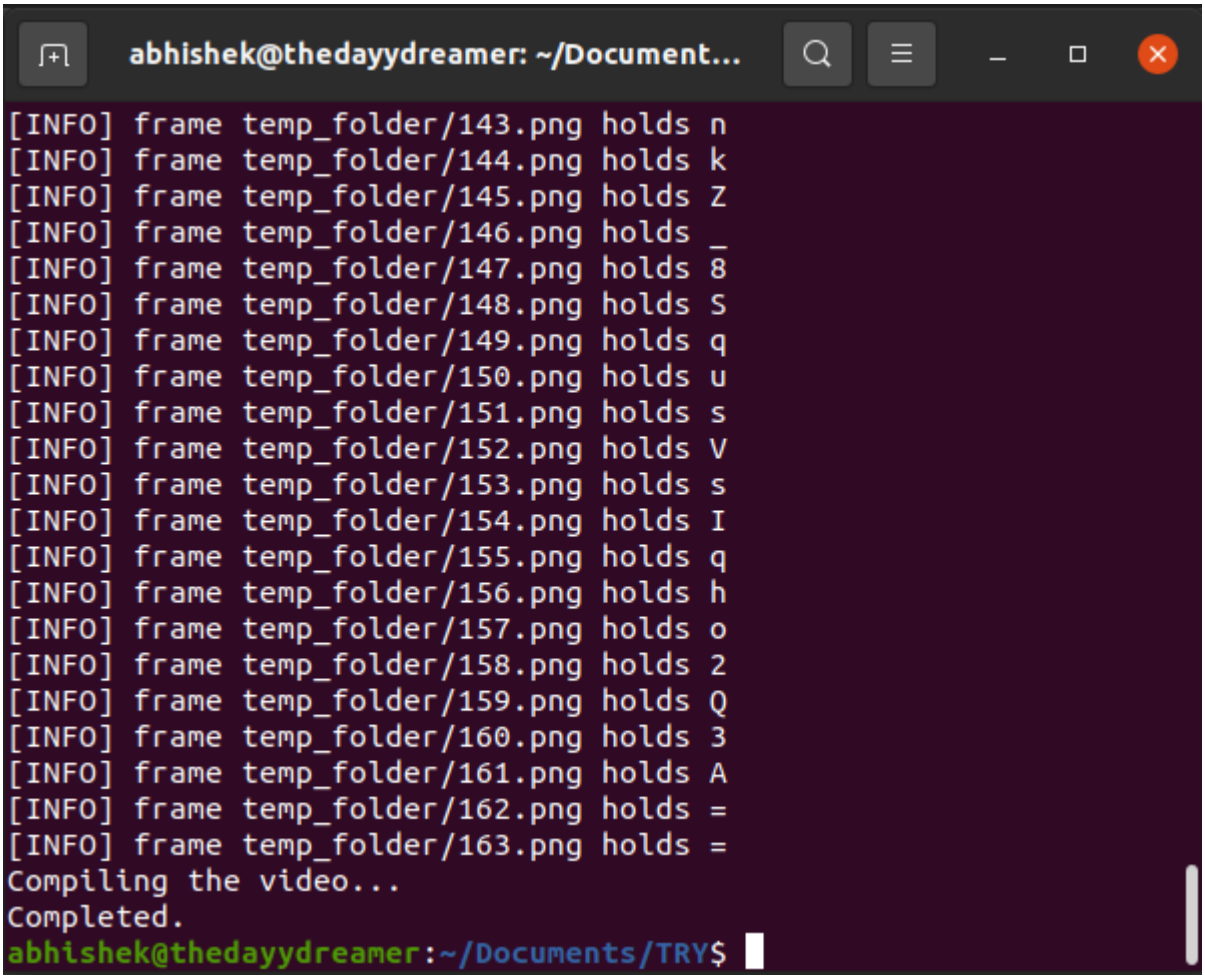


Figure 6- Encoding of data in frames 4

In the above figures (encoding of the data in frames) the secret data that is enter by user is being encoded.

The data (Image/Video pixels) is divided into each pixel that are present in the inputted data.

And the secret message is placed one bit in each pixel (LSB).

The data is placed at the least significant bit (LSB) in the pixel.

As we can see in the figure-3 to figure-6.

For the above video a total number of 163 frames are created and I each frame a bit size data is encoded.

4. Decoding

In this part of the system the encoded message that is received from the encoded module is being decoded.

Decoding of the encoded message is done with the help of the below code.

The Code:

```
import cv2

import os

import base64

from base64 import decode

from stegano import lsb

from subprocess import call, STDOUT

from pathlib import Path

from cryptography.fernet import Fernet

from cryptography.hazmat.backends import default_backend

from cryptography.hazmat.primitives import hashes

from cryptography.hazmat.primitives.kdf.pbkdf2 import PBKDF2HMAC
```

```
print("Cryptography Based Image/Vide Steganography for more secure data hiding")

print("DECODING MODULE")

print("1.Reveal Message from Image")

print("2.Reveal Message from  Video")
```

```
choice = input()



if choice == '1':
    print("Enter Image Path")
    img_path = input()
    print("Extracting the message..")
    str = lsb.reveal(img_path)
    print(str)



elif choice == '2':
    print("Checking Encryption-key")
    file = open('Encryption_key.key','rb')
    key = file.read()
    file.close()

    print("Extracting frames from the video...")

    vidcap = cv2.VideoCapture("video.mov")
    count=0
    while True:
        success, image = vidcap.read()
```

```
    if not success:
        break
    cv2.imwrite(os.path.join("temp_folder2", "{:d}.png".format(count)), image)
    count += 1

print("Extracting the message from frames...")

secret=[]
str=""
root="temp_folder2/"
for i in range(len(os.listdir(root))):
    f_name="{}{}.png".format(root,i)
    secret_dec=lsb.reveal(f_name)
    if secret_dec == None:
        break
    secret.append(secret_dec)

str =''.join([i for i in secret])
# print(''.join([i for i in secret]))
# print(str)
print("Cipher-text obtained:")
print(str)
print("Decrypting the message...")
f = Fernet(key)
```

```
    str = bytes(str, 'utf-8')
    decrypted = f.decrypt(str)
    decoded = decrypted.decode()

    file = open('Decrypted_message.txt','w+')
    file.write(decoded)
    file.close()
    print("Hidden Message: "+decoded)
    print("Completed.")
else:
    print("Invalid Choice!")
```

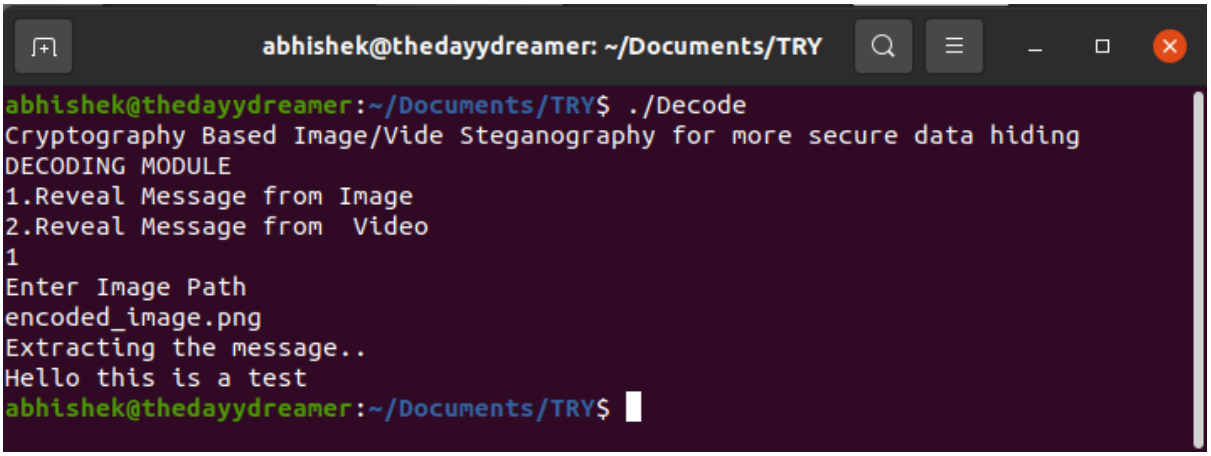


Figure 7- Secret Message from Image

After the successful execution of the above decoded code the above output screen is obtained.

So here are two option available to the user that

1 Reveal Message from Image

2 Reveal Message from Video

So according to the input data (image/video) the user should select from both option.

If the secret message is encoded with help of image than press 1 and than enter.

Otherwise, if the encoding is done on the video than press 2 and then enter

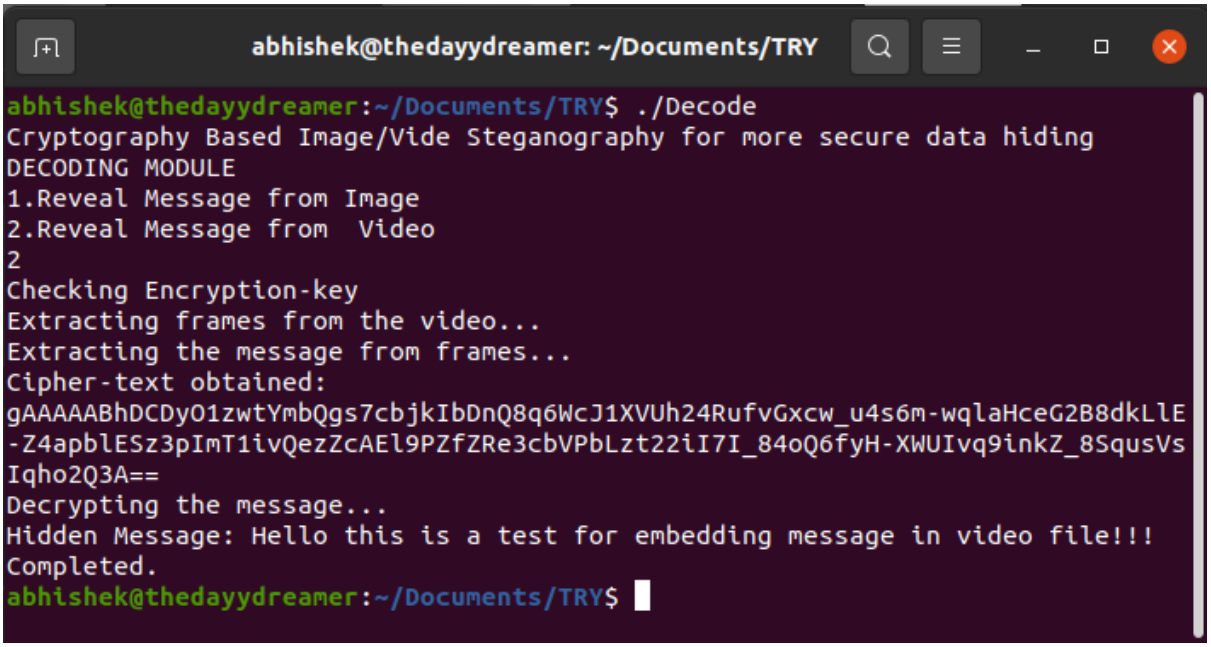


Figure 8 – Secret Message from Video

Now, further the system will try to decode the secret message from the encoded message.

For decoding purpose, the video encoded with the secret message is separated into the frames.

For decoding we will require a decryption key to decrypt the data.

And the frames that are separated, now one by one the system from each frame will retrieve the data from the place of least significant bit (LSB).

The bit data that is retrieve from the frames.

Is now will be joined in a way the required secret message can be achieved.

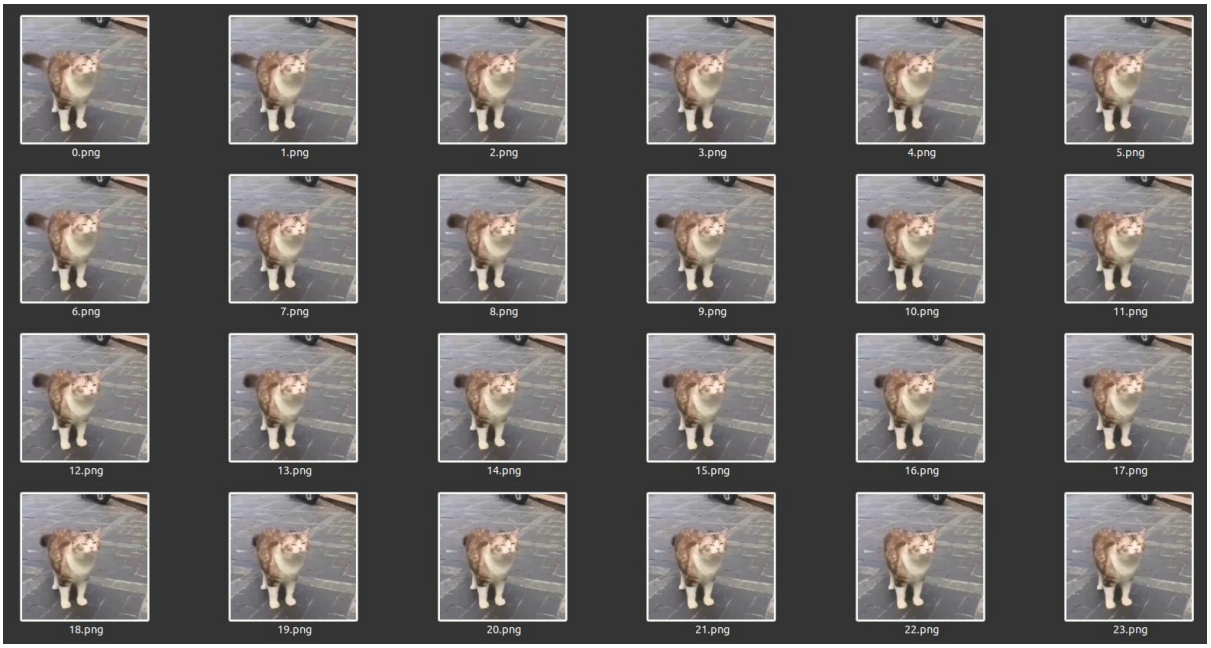


Figure 9 – Extracted Frames

In the above figure the extracted frames that are produced by decoding of the data.

These same frames are used for retrieve of the secret message.

The data that is received from here will be in cipher text form which is to be converted into normal form.

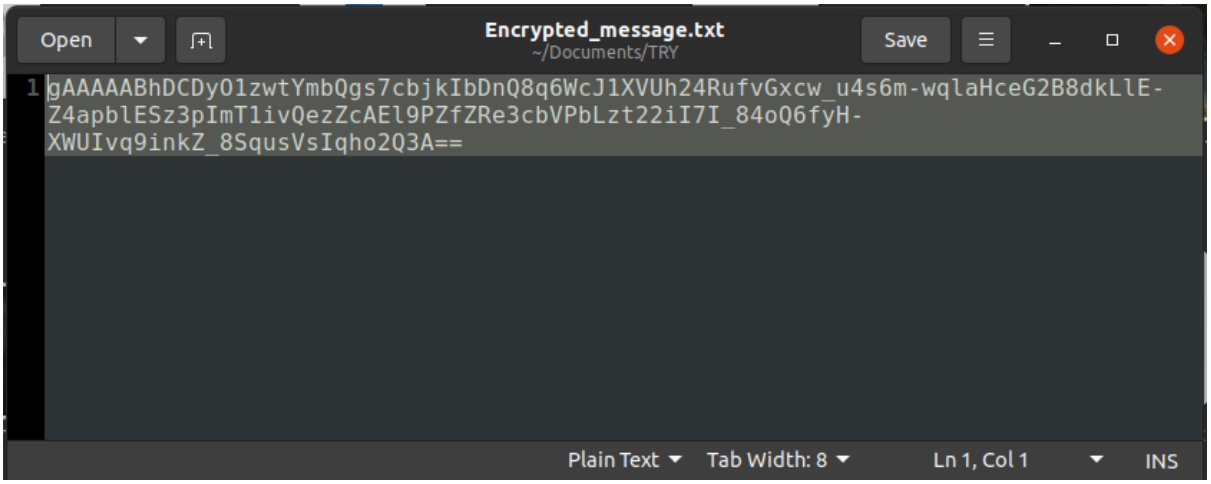


Figure 10 – Cipher Text

Cipher text is the text data that is actually saved into the bits of each frame.

And when this text is obtained.

The next step is to change this data into the form of readable text.

This cipher text is further converted into secret message at the receiver side.

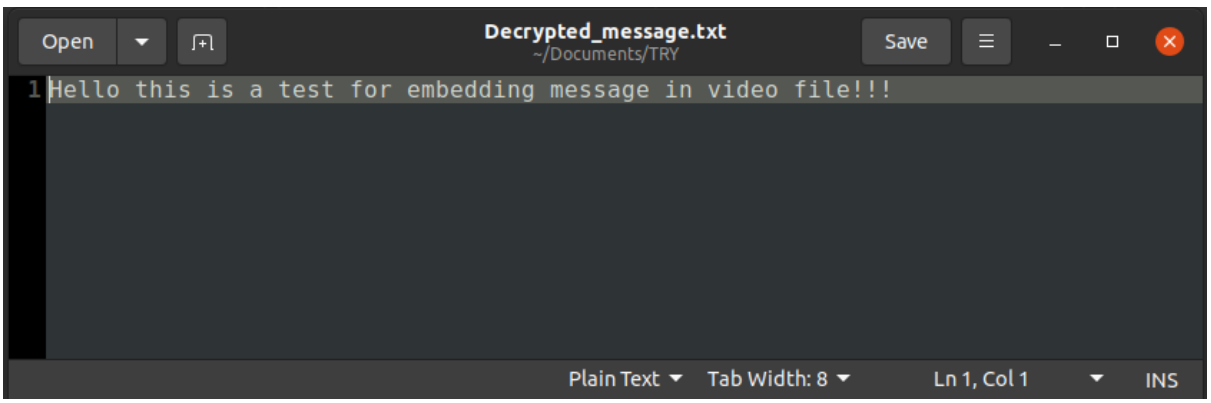


Figure 11- Decrypted Message

In the above figure the required secret message is being obtained after successful execution of the system.

This Decrypted message is achieved after conversion of the cipher text.

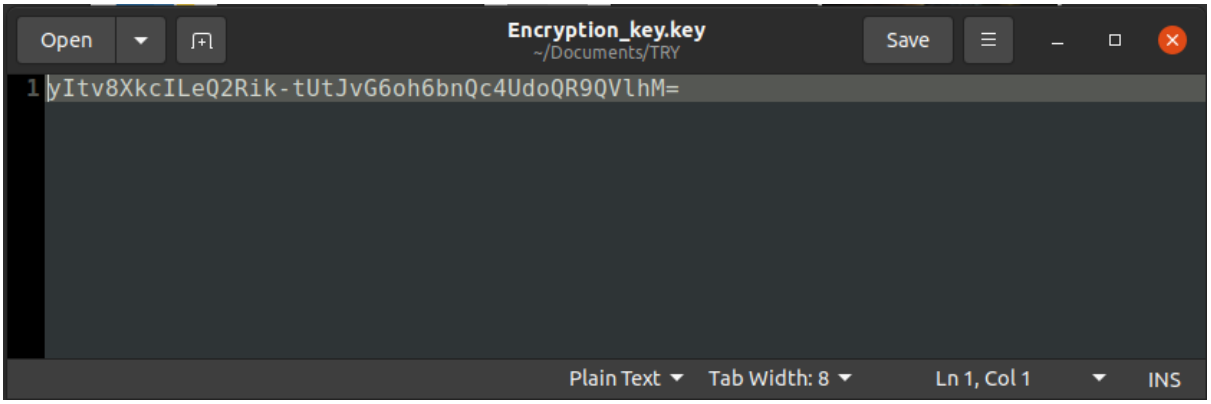


Figure 12- Encryption Key

Encryption Key is used as a password that is placed into the encoded message.

And without this key its not possible for anyone to decode the encoded message.

So, it in a way ensures the secret of the message.

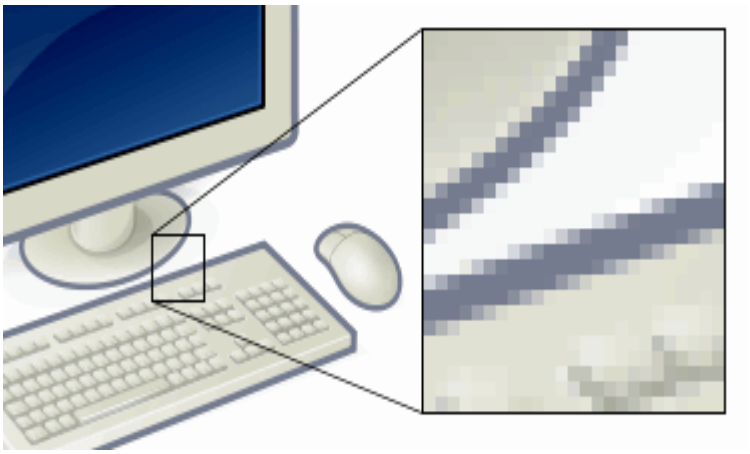
7. IMPLEMENTATION

How to hide data in image/video?

LSB (Least Significant Bit) METHOD

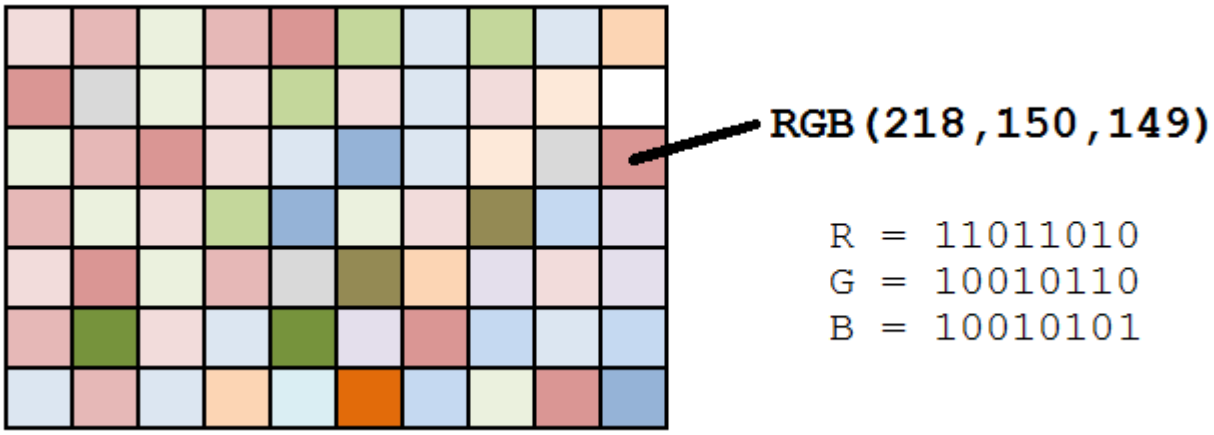
What is a digital image?

We can describe a digital image as a finite set of digital values, called pixels. Pixels are the smallest individual element of an image, holding values that represent the brightness of a given color at any specific point. So we can think of an image as a matrix (or a two-dimensional array) of pixels which contains a fixed number of rows and columns. which are basically a dot matrix data structure, representing a grid of pixels, which in turn can be stored in image files with varying formats.



So, each pixel from the image is composed of 3 values (red, green, blue) which are 8-bit values (the range is 0–255).

As we can see in the image, for each pixel we have three values, which can be represented in binary code (the computer language).



	128	64	32	16	8	4	2	1
8 bit binary digit	1	0	1	1	0	0	0	1
	128 + 32 + 16 + 1 = 177							

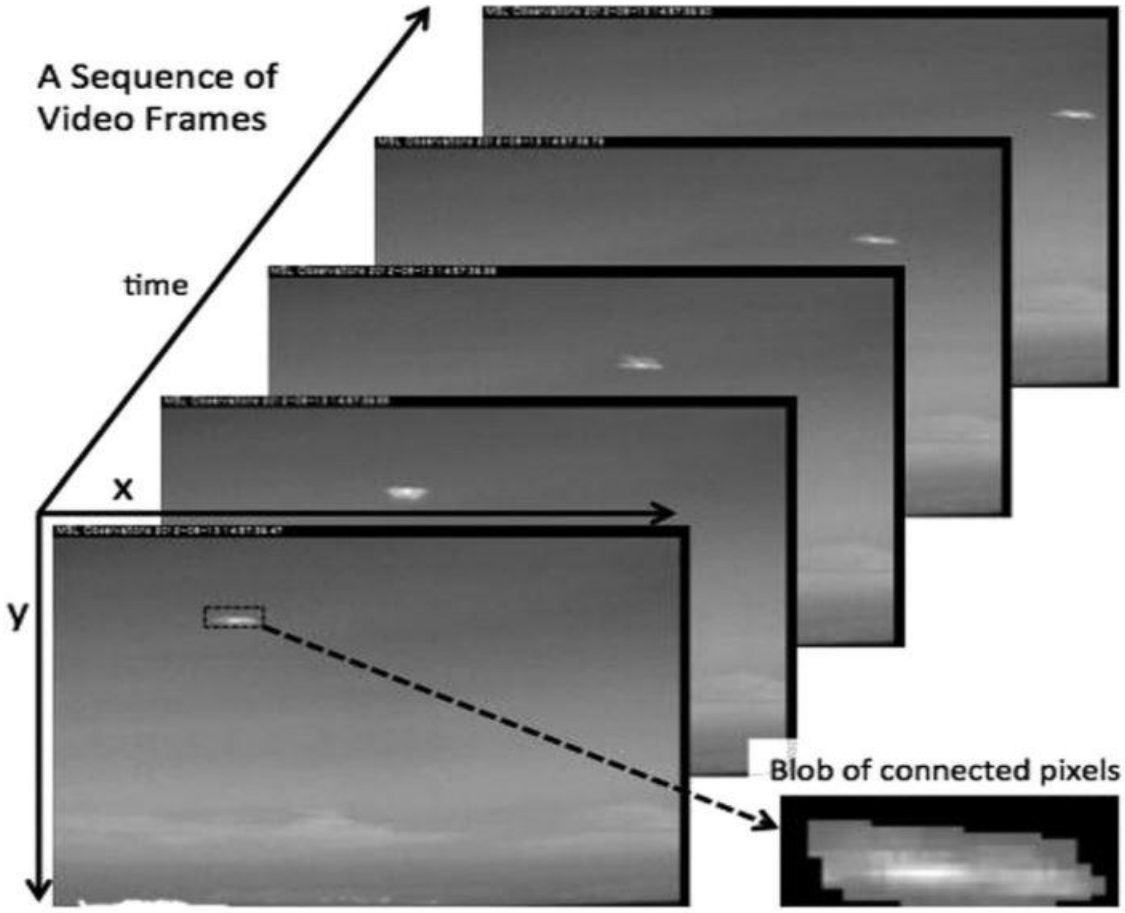
When working with binary codes, we have more significant bits and less significant bits, as you can see in the image below.

The leftmost bit is the most significant bit. If we change the leftmost bit it will have a large impact on the final value. For example, if we change the leftmost bit from 1 to 0 (11111111 to 01111111) it will change the decimal value from 255 to 127.

the rightmost bit is the less significant bit. If we change the rightmost bit it will have less impact on the final value. For example, if we change the leftmost bit from 1 to 0 (11111111 to 11111110) it will change the decimal value from 255 to 254. Note that the rightmost bit will change only 1 in a range of 256 (it represents less than 1%).

What is a video?

A video is a sequence of images called frames. Each frame is a two-dimensional grid of pixels or single image. A video is a collection of frames, and each frame is an image. So if we pull out all the frames from a video, we can use this method to store our data using LSB steganography and stitch those frames back into a video with the secret message.



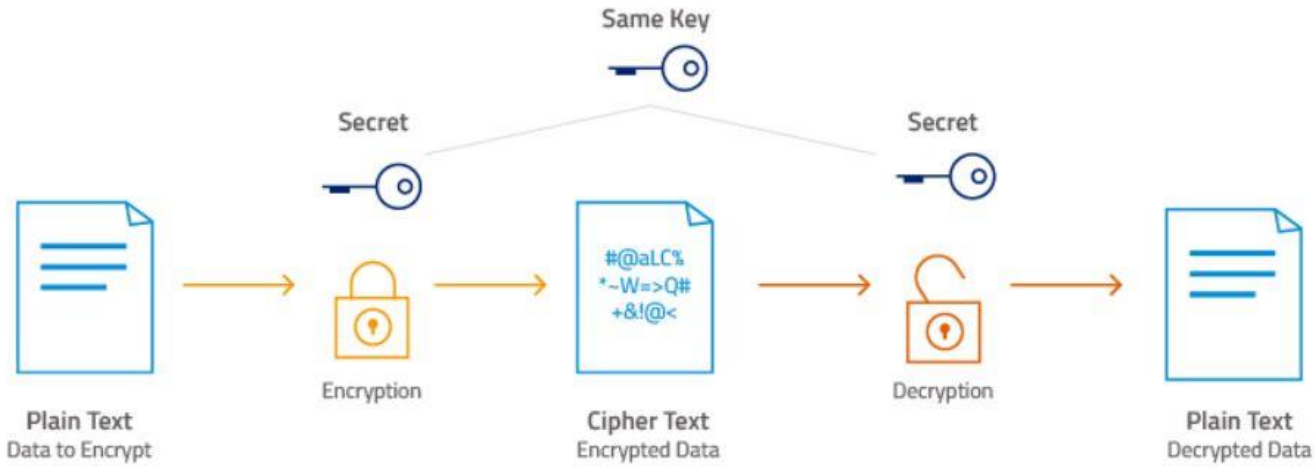
Encrypting the data for more secure data hiding:

What is Encryption?

Encryption is a way of scrambling data so that only authorized parties can understand the information. In technical terms, it is the process of converting human-readable plaintext to incomprehensible text, also known as ciphertext. In simpler terms, encryption takes readable data and alters it so that it appears random. Encryption requires the use of a cryptographic key: a set of mathematical values that both the sender and the recipient of an encrypted message agree on.

AES-256 encryption:

AES-256, which has a key length of 256 bits, supports the largest bit size and is practically unbreakable by brute force based on current computing power, making it the strongest encryption standard.



How does AES 256 work?

AES is a symmetric key cipher. This means the same secret key is used for both encryption and decryption, and both the sender and receiver of the data need a copy of the key.

The data to be encrypted (known as plaintext) is divided into sections called blocks. AES uses a 128-bit block size, in which data is divided into a four-by-four array containing 16 bytes. Since there are eight bits per byte, the total in each block is 128 bits. The size of the encrypted data remains the same: 128 bits of plaintext yields 128 bits of ciphertext.

The basic principle of all encryption is that each unit of data is replaced by a different one according to the security key. More specifically, AES was designed as a substitution-permutation network. AES brings additional security because it uses a key expansion process in which the initial key is used to come up with a series of new keys called round keys. These round keys are generated over multiple rounds of modification, each of which makes it harder to break the encryption.

Fig : Encryption and Decryption

Implementation using python:

The lsb-method steganography and cryptography is implemented using python because of its vast library and support of developers across the globe

Easy to implement

We have divided the project into two parts:

1. Encoding (or Sender module)
2. Decoding (Or receiver module)

Encoding module:

Procedure for Image Steganography:

1. Run the encode.elf (linux executable) on terminal
2. A drop-down menu is given

3. Select from the given options Image or Video

```
abhishek@thedayydreamer: ~/Documents/TRY
abhishek@thedayydreamer:~/Documents/TRY$ ./Encode
Cryptography Based Image/Vide Steganography for more secure data hiding
ENCODING MODULE
1.Hide Message in Image
2.Hide Message in Video
1
Enter Image Path
image.png
Enter Message
Hello this is a test
Encoding...
Completed.....
abhishek@thedayydreamer:~/Documents/TRY$
```

RESULT:

Original Image

Embedded Image

Message/data extraction

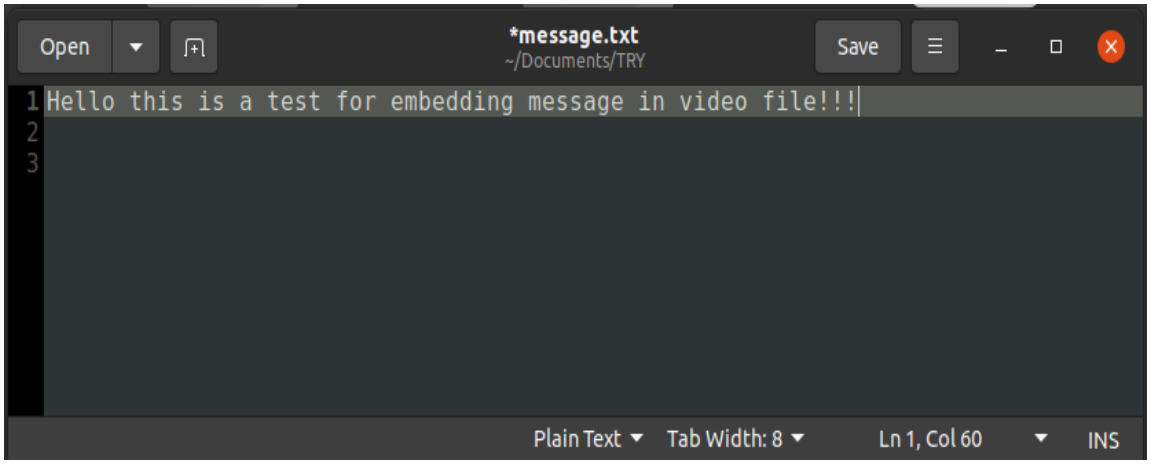
Procedure for message extraction from image

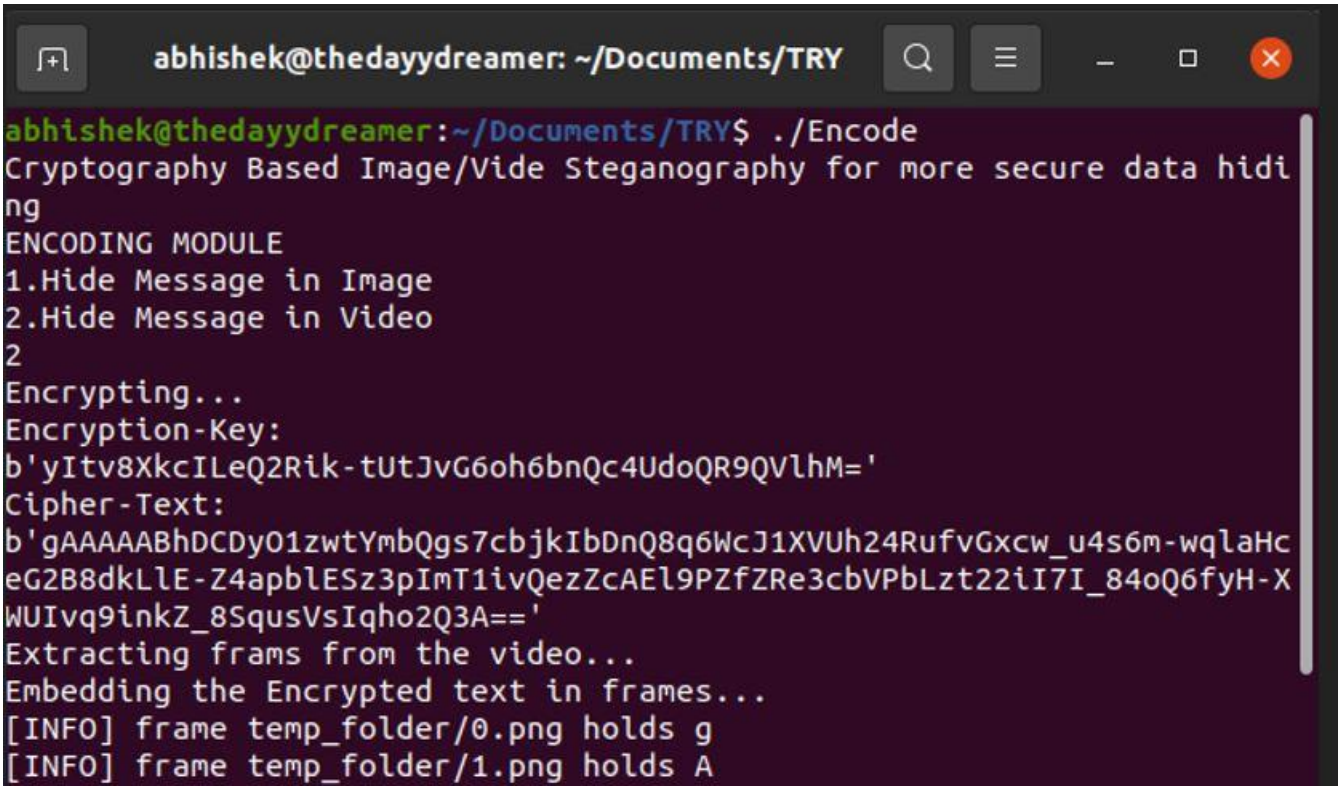
1. Run the decode.elf (decoding module)
2. Select the Image option from drop-down menu

```
abhishek@thedayydreamer: ~/Documents/TRY
abhishek@thedayydreamer:~/Documents/TRY$ ./Decode
Cryptography Based Image/Vide Steganography for more secure data hiding
DECODING MODULE
1.Reveal Message from Image
2.Reveal Message from  Video
1
Enter Image Path
encoded_image.png
Extracting the message..
Hello this is a test
abhishek@thedayydreamer:~/Documents/TRY$
```

Procedure for video steganography

1. Run the encode module again
2. Select video option from drop down menu
3. Save the message in a file named message.txt
4. Select the video

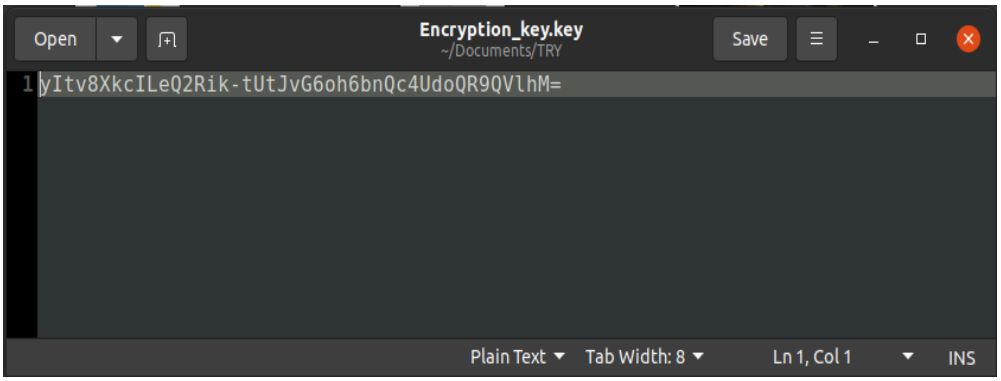


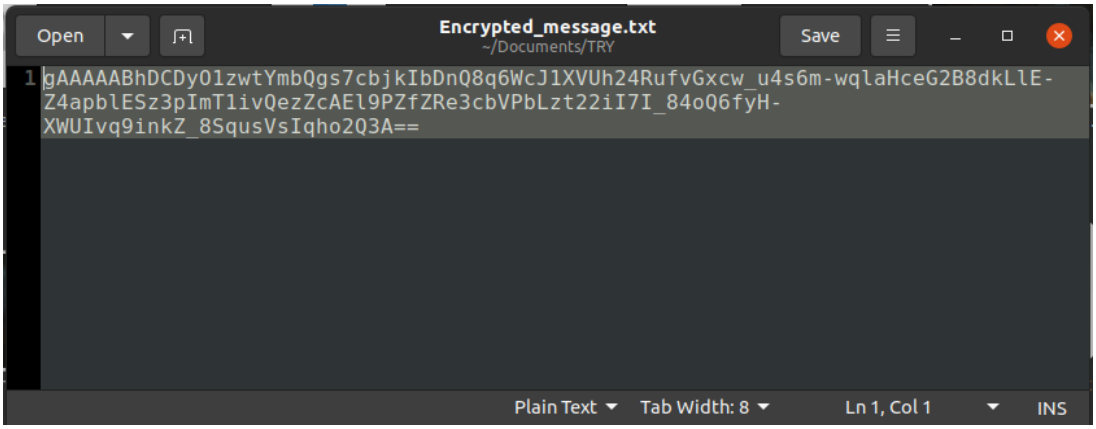


An Encryption key file is generated “encryption_key.key”

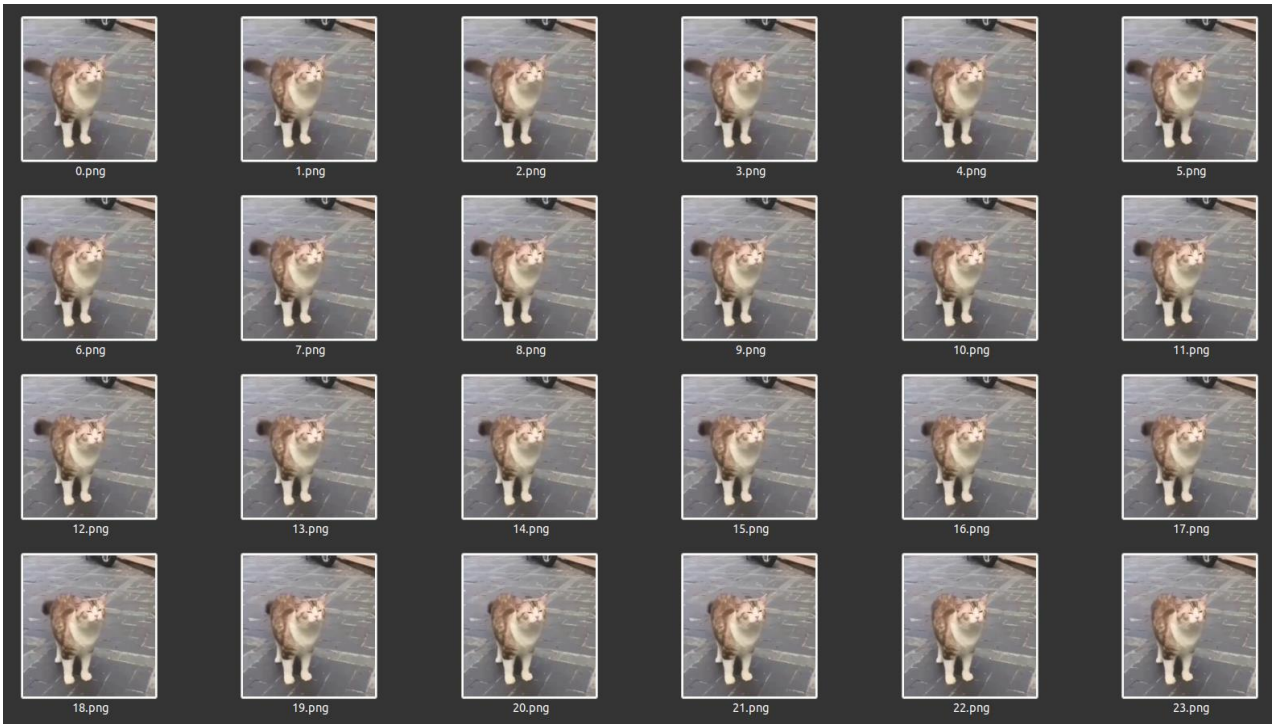
This encryption key is essential for message extraction

With the help of this key a encrypted message is generated with is then further embedded into the video file.




Open
Encrypted_message.txt
~/Documents/TRY
Save
1 gAAAAABhDCDyO1zwtYmbQgs7cbjkIbDnQ8q6WcJ1XVUh24RufvGxcw_u4s6m-wqlaHceG2B8dkLlE-
Z4apblESz3pImT1ivQezZcAEl9PZfZRe3cbVPbLzt22iI7I_84oQ6fyH-
XWUIvq9inkZ_8SqusVsIqho2Q3A==
Plain Text
Tab Width: 8
Ln 1, Col 1
INS

Frames Extraction and message embedding:



Extracted Frames

```
[INFO] frame temp_folder/0.png holds g
[INFO] frame temp_folder/1.png holds A
[INFO] frame temp_folder/2.png holds A
[INFO] frame temp_folder/3.png holds A
[INFO] frame temp_folder/4.png holds A
[INFO] frame temp_folder/5.png holds A
[INFO] frame temp_folder/6.png holds B
[INFO] frame temp_folder/7.png holds h
[INFO] frame temp_folder/8.png holds D
[INFO] frame temp_folder/9.png holds C
[INFO] frame temp_folder/10.png holds D
[INFO] frame temp_folder/11.png holds y
[INFO] frame temp_folder/12.png holds O
[INFO] frame temp_folder/13.png holds 1
[INFO] frame temp_folder/14.png holds z
```

Message/Data Embedding

8. Requirements

8.1 Functional Requirements

Functional requirements are the requirements that define specific behavior or function of the system.

• Secret Text Message File: In this file you will have to write secret message to hide or you can select any text file of secret message.

• Cover Image: Cover Image is the image is to be selected in which secret text message can be hidden.

• Stego Encryption LSB implementation is performed on cover image to hide secret text message by replacing bits of cover image by the bits of message. • Sender In this Sender send this stego image file to intended recipient to which he does want to communicate.

• Receiver In this receiver receives the stego image and opens in decryption option for getting hidden text message inside that image.

8.2 Non-Functional Requirements

• Safety Requirements:

Sender and Receiver should make sure that only they are having the same software to encrypt and decrypt data inside image. Both should take care of eavesdropping.

• Security Requirements:

We are going to develop a software in which embedding secret text data in image.Only sender and receiver should be aware of encrypted file. User should not unfold the message regarding sent image as well as receiver information.

• Software Quality Attributes:

The Quality of the software is maintained in such a way that only sender and receiver can communicate through image.There is no probability of knowing secret image.

8.3 System Requirements

8.3.1 Software Requirements

• Operating System:Windows 10.

• Tool:Netbeans.

• PhotoshopCC.

• Python 3

• Python Libraries used:

- Stegano
- Fernet
- Base64
- Opencv as cv2
- Pathlib
- Cryptography

8.3.2 Hardware Requirements

• INTEL I5 2.50 GHZ 4 GB RAM

• Minimum Hardware Requirement: Pentium 3 166 MHZ Or Higher 128 mb RAM

Conclusion

- Steganography works by hiding information in a way that doesn’t create suspicion.
- Lsb (least significant bit) method was used as it changes the least pixel information in the source file.
- Cryptography is added for further security and integrity of the message/data.
- Steganography, the practice of hiding information, has been around for centuries. And in parallel to technological advances, steganography has also evolved and adapted with the advent of computers and the internet.

References

Web Help:

- WWW.wikipedia.org/
- www.researchgate.net/
- www.britannica.com/
- www.academia.edu/
- https://itnext.io/steganography-101-lsb-introduction-with-python-4c4803e08041
- https://towardsdatascience.com/hiding-data-in-an-image-image-steganography-using-python-e491b68b1372
- https://towardsdatascience.com/steganography-hiding-an-image-inside-another-77ca66b2acb1
- https://www.ijltet.org/wp-content/uploads/2015/02/60.pdf

Research Papers on steganography:

- A Review Paper on Video Steganography IEEE, 2016 by Dr. Arjun Nichal
- Video Steganography: Secure Data Hiding Techniques I. J. Computer Network and Information Security, 2017
- Colour Image Steganography Based on Pixel Value Differencing in Spatial Domain, IEEE, 2018