Tribhuvan University
Faculty of Humanities and Social Sciences

“aspHide”

A PROJECT REPORT

<u>Submitted To</u>

Department of Computer Application

Mahendra Morang Adarsha Multiple Campus

In partial fulfillment of the requirements for Bachelors of Computer Application

<u>Submitted By</u>

Niroj Poudel (6-2-003-0111-2019)

November 2023

Under the Supervision of

Mr. Ujjwal Rijal

Tribhuvan University

Faculty of Humanities and Social Sciences

Mahendra Morang Adarsha Multiple Campus

Supervisor's Recommendation

I hereby recommend that this project prepared under my supervision by Niroj Poudel entitled **"aspHide"** in partial fulfillment of the requirements for the degree of Bachelors of Computer Application is recommended for the final evaluation.

____

Signature

Mr. Ujjwal Rijal

Project Supervisor

Department of BCA

Mahendra Morang Adarsha Multiple Campus

Tribhuvan University

Faculty of Humanities and Social Sciences

Mahendra Morang Adarsha Multiple Campus

LETTER OF APPROVAL

This is to certify that this project prepared by Niroj Poudel entitled "aspHide" in partial fulfillment of the requirements for the degree of Bachelors of Computer Application has been evaluated. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

__________________	__________________
Mr. Ujjwal Rijal	Mr. Kiran Ghimire
Project Supervisor	Program Director
Department of BCA	Department of BCA
Mahendra Morang Adarsha Multiple Campus	Mahendra Morang Adarsha Multiple Campus
__________________	__________________
Internal Examiner	External Examiner

Abstract

A steganography is a process of hiding the text or messages in a non-secret objects like images, audio, visuals. This specific method of encryption was started from ancient Greece. In today's age of information this method is being used by many people as a secret means of communication. This method of hiding the data in plain sight was one of the most popular way of encryption during the world war, before digital encryption was widely implemented.

It is not so easy to share a confidential message across the internet without the fear of being interception of the credentials transferred. Many countries like Iran, Cuba, Sudan, Syria do not allow SSL which leads to the insecure communication over the internet. Content shared via internet can be intercepted and disclosed by various attacks like man in the middle and packed sniffing. This will lead to zero confidentiality in the digital world. So, to solve the issue aspHide is developed.

With aspHide users can prepare a secret message hidden in the image and share it with any person. Even if the shared data is intercepted in between actual data hidden in the image will not be visible to the attacker. For additional security of the hidden message the data is encrypted with a user provided password before hiding it into the image and receiver needs the password to decrypt the message. Even if there is no security at all, the credentials of yours will reach the other end in a safe and secure manner.

Acknowledgement

Foremost, I would like to express my sincere thanks of gratitude to my advisor, Mr. Ujjwal Rijal for all his continuous support and help in the project. I cannot thank him enough for all his patience, motivation and immense knowledge he provided during the difficult days. I could not have imagined the project without the guidance and mentorship provided by him. Beside our advisor, we would like to thank our program coordinator Mr. Kiran Ghimire who allowed the project to continue and gave this immense opportunity to work on this wonderful project topic “aspHide”. Furthermore, they also provided us the platform for the required research beyond our academics and assisted, coordinated as a team to let the project meet its optimum height. I am really thankful to them for their support, help, guidance, motivation and corrections. I appreciate all the technical support and motivation given by BCA program of Mahendra Morang Adarasha Multiple Campus and all the help provided in order to keep this project aligned with the actual; objectives.

Lastly, I would also like to thank my parents, colleagues and every supportive person who have helped me a lot in finalizing this project within the limited time frame. Without the support and help you have provided, I would never have stepped into this project.

I have tried to mention and give credit to everyone who have helped me in the project, along with the sources from where I collected the required data and information which supported this project. Yet, there may be some unintended errors and some sources or individuals may have been missed to mention. I shall feel obligated if they are brought to my notice.

Authors
Niroj Poudel
October, 2023

List of Abbreviations

Terms	**Meaning**
DFD	Data Flow Diagram
IDE	Integrated Development Environment
UI	User Interface
TBI	Text Based Interface
SSL	Secure Socket Layer
TLS	Transport Layer Security
EXIF	Exchangeable Image File Format
LSB	Least Significant Bit

List of Figures

List of Tables

Table of Contents

Chapter 1: Introduction

1.1. Introduction

Project aspHide is created as a tool for secret communication among people where data privacy is denied by the government or any form. It can be very useful in the places where cryptography in internet is not allowed to be used. In such places, if you want to have a secure communication, you can use this tool that will take a message from users, encrypt it first and hide it into the provided image. The tool than prepares a new image with the encrypted message inside. The quality change in image, or the hidden message won't be visible to any naked eye nor will the hidden message will be detected by any tool or technology, used to filter the internet traffic.

This tool is based on python and has a text based interface (TBI), which is super simple to use and to work upon by any people. This tool needs no any deep knowledge or understanding about any technology. User simply need to provide image, message and password than the tool does the task of encrypting and hiding the encrypted text into the image.

1.2. Problem Statement

Internet is the new place to go, when we need anything. Our friends, entertainment, grocery, cinema and our daily life is in internet and it need to be secure. Modern day security is being provided by SSL and TLS but many countries like Cuba, Iran, Syria and Sudan the standard internet securities is intentionally removed by government [1]. So, encryption over internet is illegal in those countries. Whenever you have to communicate among your friends or anyone, the internet traffic is not secure. Various attackers or even worse the government can harvest the citizen's personal communications which may be used against them.

So, in those places or even in secure places, if you want to share confidential data like organization server password, other credentials or anything you don't want other to see, will always be in risk of being broadcasted by hackers. This tool solves the problem, as the messages are encrypted in local machine and hidden in casual image that you have captured or downloaded from web. This tool hides your secret message in an ordinary image file which if even intercepted, has nothing to be worried about.

1.3. Objectives

- To provide a means of secret communication to those people who want to share or have a secret communication over the internet in insecure environment.
- To protect the information privacy of the people in the internet where standard encryption technology like SSL cannot be used.

1.4. Scope and Limitations

Scope: Project aspHide is a terminal based application which is programmed mainly using Python is a steganography tool. Its main aim is to implement the CIA triad of security where there is an unsafe cyber space for normal communication.

- Make user feel safe and secure while sharing their personal data over the internet.

Limitations:

- Requires internet connectivity
- User must be digitally literate
- Does not support windows operating system

1.5 Report Organization

On completion of our project development, we have documented the milestone and the final document report has been organized under the following chapters.

Chapter 1 describes the overall view of the project, the basic problem definition and the general overview of the problem which describes the problem in layman terms, It also specifies the software used and proposed solution strategy.

Chapter 2 describes the fundamental theories, general concepts and all the terminologies that are related to the project. Whereas in literature review section we review the similar projects done by other and what conclusion they find out by the project. We collect the theories paper and review their document.

Chapter 3 describes the functional and non-functional requirements for the smooth operation of the application. It describes whether the system that is designed is feasible or not. It also contains technical diagrams like the DFD and System Architecture. Furthermore the applied logics and algorithm are described here in details.

Chapter 4 describes the different technologies used for the entire development process of the user interface, business logic and the workflow of the entire project. It describes the tools that are used to develop the application. How modules are developed and tested, the test cases and results are also published in this chapter.

Chapter 5 has the theme and self-review regarding the developed system. It also has some screenshots of the projects to prove the test cases and its basic functionalities. More than that this section includes the lesson learnt during the project, its outcome and the plan on how it will be updated and changed in coming days. Finally, this section is concluded with a conclusion of the project and future recommendation.

Chapter 2: Background Study and Literature Review

2.1 Background Study

Since the enhancement of technology and process of digitization has changed the process of doing many jobs, making the job simpler, efficient and reliable than ever. Every step towards the simplicity or the automation of the task over internet brings threat to privacy. Steganography tool used to be a popular secret communication method during the pre-internet phase. The simplicity of the algorithm makes it super handy to develop and modify as per the need of current environment. There are many tested and implemented methods for steganography, but the most popular and simplest one is the image steganography. In this method the colors codes of the image are altered slightly to embed the message within. Previously the cryptographer used to utilize the EXIF data of the image to hide message, as technology improved it got less secure, so now methods like LSB algorithm is used to gain the same result with more resilience. [2]

Here, in the system, the LSB algorithm is slightly modified to gain the efficiency of flow of the program for better time complexity. The program only utilizes the red channel of the RGB color and leave others unchanged. This program has the image steganography feature with the AES using a PSK.

2.2 Literature Review

Privacy and digitization should go along with each other, so to obtain that we must be concerned with the CIA triad while sharing our confidential information. But it's not like we have every control over our data in the internet. So, at least while sharing the data with others we can adapt some measures to be safe. For the purpose, I have reviewed some pre-existing systems.

SteganPEG is a lightweight software application built specifically for helping users hide sensitive data inside JPG images in only a few steps. It sports a clean and simple interface that offers only a few configuration settings to tinker with. The program gives you the possibility to embed photos into the working environment using the built-in browse function, so you cannot rely on "drag and drop" operations. SteganPEG allows you to encrypt data by setting up passwords, preview the picture in the primary panel, as well as insert multiple items to be hidden in the photos. What's more, the utility is able to indicate how much space is occupied by the hidden files so you can calculate if there's available space for adding more items. The photo, which embeds the sensitive data, can be exported to JPG file format, provided that you have specified the filename and saving directory. Since it doesn't require much computer knowledge to work with this utility, even rookies can master the entire process with minimum effort. During our testing we have noticed that SteganPEG accomplishes a task very quickly and without errors throughout the entire process. As it would be expected from such a small app, it remains light on the system resources, so it doesn't burden computer performance, nor interfere with other programs' functionality. To sum it up, SteganPEG seems to be the right choice in case you are looking for a simple-to-use encryption application that comes packed with only a few dedicated parameters, and is suitable especially for rookies. [3]

Hide'N'Send is a small utility which offers steganography. It lets you hide any kind of file behind a JPG image file. It supports hashing and encryption too. This means that you can hide your data by encrypting. This adds an extra layer of security. The interface of the tool is simple and offers two tabs — one to hide data and other to extract data. You can select the options accordingly. Just run the tool, select the image file, then select the file which you want to hide, select the encryption type and then hide the data in the image. Use the same tool again to extract the hidden information in the image. [4]

Camouflage is another steganography tool that lets you hide any type of file inside of a file. There is no kind of restriction in the software for hiding the file. Use of the tool is simple and easy: you can just right-click on any file and select the Camouflage option. To extract your sensitive data from the file, right-click and select uncamouflage. You can also set a password to encrypt the hidden data inside the file. The project is no longer in development, but you can use the old file for your work. It still performs well and you can use it to hide your confidential data inside an image. [5]

OpenStaego can hide any data within an image file. It also does watermarking image files with an individual signature. It can be used to detect unauthorized file copying. This tool has standard licensing on watermark technology and image with message may be distinguished by any people. [6]

LSB Algorithm (R-Channel) is the most common and popular method of modern day steganography is to make use of LSB (least significant bit) of picture's pixel information. This technique works best when the file is longer than the message file and if image is grayscale. When applying LSB techniques to each byte of a 24-bit image, three bits can be encoded into each pixel.

Pixels: (00100111 11101001 11001000)

(00100111 11001000 11101001)

(11001000 00100111 11101001)

A: 01000001

Result: (00100110 11101001 11001000)

(00100110 11001000 11101000)

(11001000 00100111 11101001)

LSB insertion is easy to implement, at the same time, easily attacked. Today steganography is being incorporated into digital technology. The techniques have been used to create the watermarks that are in our nation's currency, as well as encode music information in the ever-popular mp3 music file. Copyrights can be included in files, and fingerprints can be used to identify the people who break copyright agreements. While Slight modifications in the color palette and simple image manipulations will destroy the entire hidden message. [7]

Chapter 3: System Analysis and Design

3.1 System Analysis

3.1.1 Requirement Analysis

i. Functional requirements

- User will be able to navigate the program features with text based command.
- Users will be able to select a desired image to hide message in it.
- Users will be able to set a password to encrypt their message.
- Users will be able to select the desired image to extract messages within.
- Users will be able to enter the pre shared key to decrypt the encrypted message.
- Users will be able to read the decrypted message.
- Users will be able to save the decrypted message in a text file.
- Users will be able to exit the program from any point in the program without causing any damage to the image being processed.

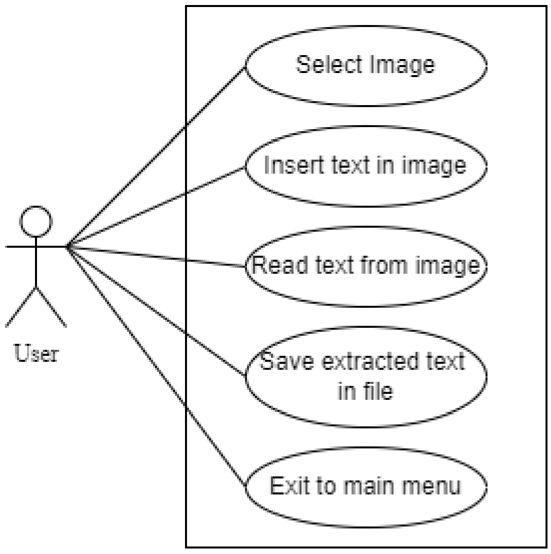


Figure 1: Use case diagram

ii. Non Functional Requirements

Some of the non-functional requirements of the system are discussed below.

- **Availability:** Once setup, the program does not require any internet connectivity to do the required task. Users can access the function that the program provides from any supported devices they wish. This is totally a standalone program so there are no any pre requisites.

- **Security:** The program does not store the information about the data that the user is sharing via the image. Also the data to be shared uses AES before hiding it, so the chances of discovering the message within the image is very low. Top to this, the password being shared is also hashed beforehand.
- **Performance:** Program uses python and its modules to perform the required task. Since python uses pycache technology for efficiency, developed system does not have any performance issue in terms of time complexity. Also the program uses the least amount of memory as possible.
- **Reliability:** The algorithm is well studied and implemented to provide the best for the users. There is no any known loopholes in the algorithm itself. And on top of that the program uses the AES, which is one of the best choice for many top companies for the encryption. So the program is reliable in terms of output and security.

3.1.2 Feasibility Study

Following feasibility study were done before building the system to see if the system could be built with exact requirements in required time.

i. Technical Feasibility

The system being developed is based of python and its modules. The technology is widely supported by any computing system the world. The basic requirements like ARM Cortex or Pentium 4, 500 MB of RAM, 5 GB of storage are too common in this era. So the technology to develop the program is usable in the current situation.

ii. Economic Feasibility

The program does not require any sort of proprietary software or hardware that may increase the cost if production of software. The product after being ready will be available as a free software to every user. The technology used here are open source and no cost is required to implement it. In long run we can even integrate some premium features in it, which may also support the program to develop and integrate some proprietary technology.

iii. Operational Feasibility

The developed system has a text based interface, like of the arcade games of 2000s’. The interface is super simple and efficient as it shows the hint of the input to be given. Every user with the supported device can freely and uninterruptedly use the features available using the navigation provided.

iv. Schedule Feasibility

For this feasibility study have estimated the time duration of each step in the development process and plotted it into a Gantt-chart for visualizing the process. From the below table we can conclude the project is feasible as per the schedule.

Table 1: Gantt Chart

Task\Weeks	1	2	3	4	5	6	7	8	9	10	11	12
Planning	■	■										
Analysis		■	■	■	■							
Designing				■	■	■	■	■				
Implementation							■	■	■	■		
Testing										■	■	
Documentation								■	■	■	■	■

3.1.3 Process Modeling

The process flow of the project is demonstrated below in the data flow diagram.

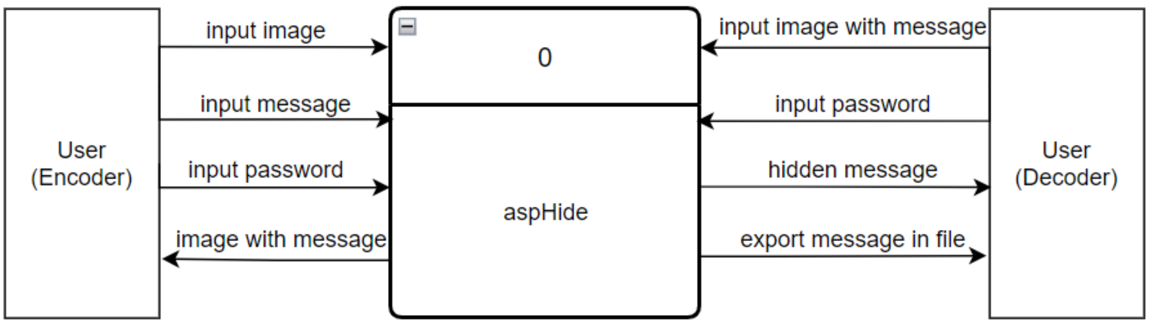


Figure 2: Context diagram of aspHide

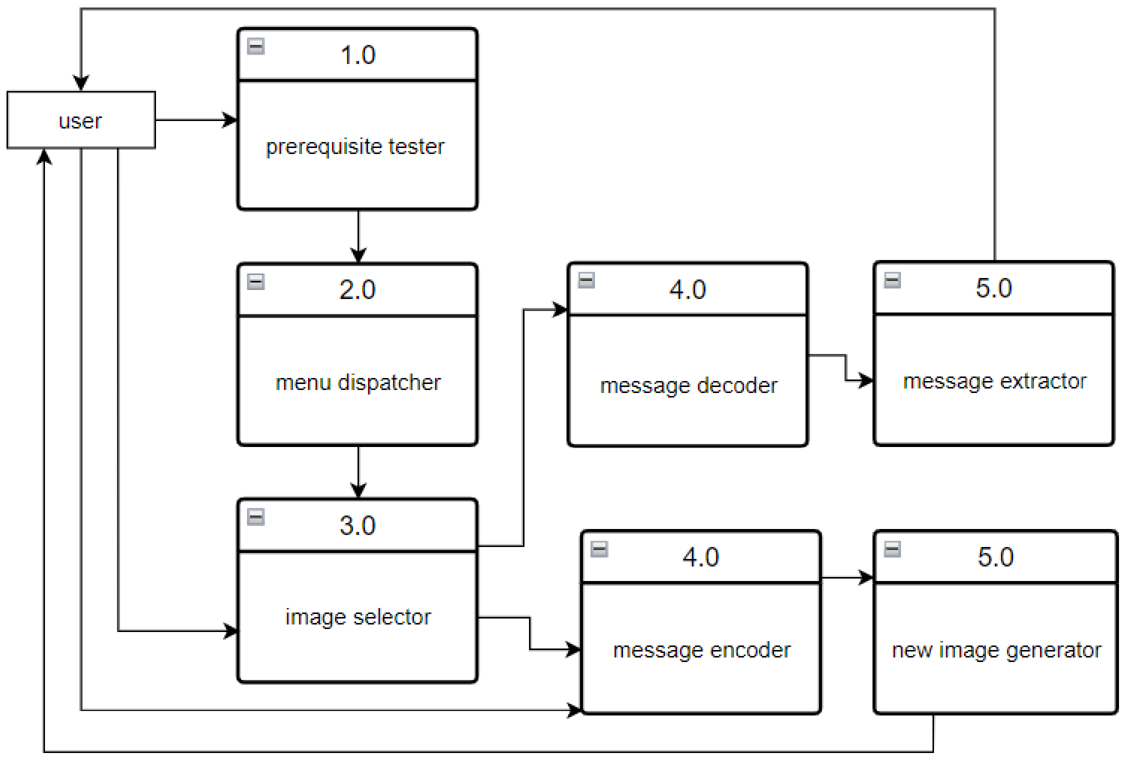


Figure 3: DFD of aspHide

3.2 System Design

3.2.1 Architectural Design

The below architectural diagram shows how the system works and it is demonstrating the role of users in the developed program.

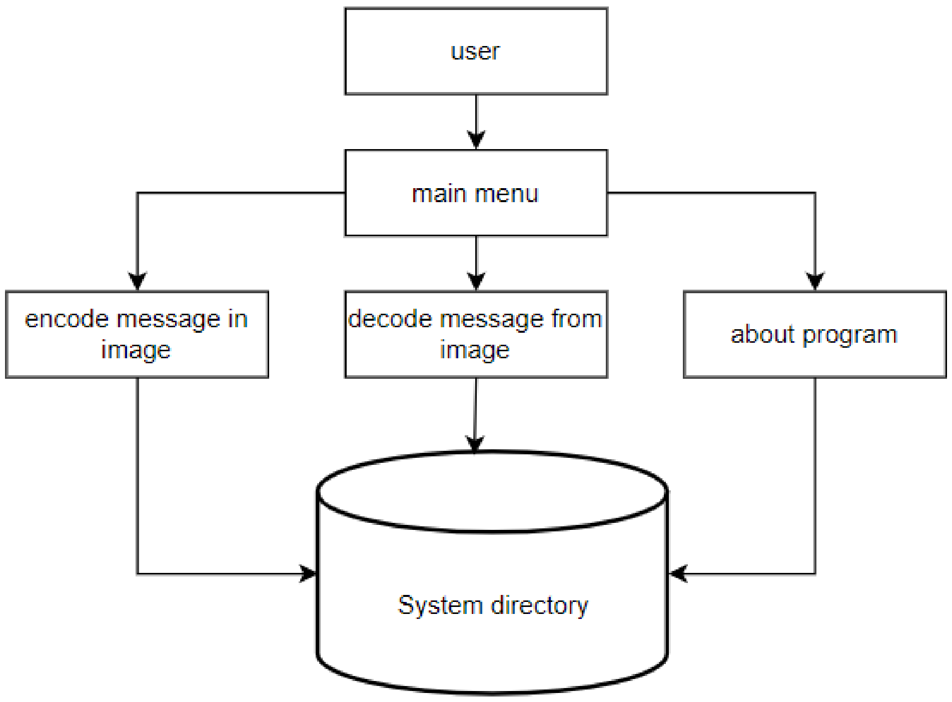


Figure 4: Architecture diagram of aspHide

3.2.2 Interface Design

The program has a simplistic text based interface that follows following interface design.

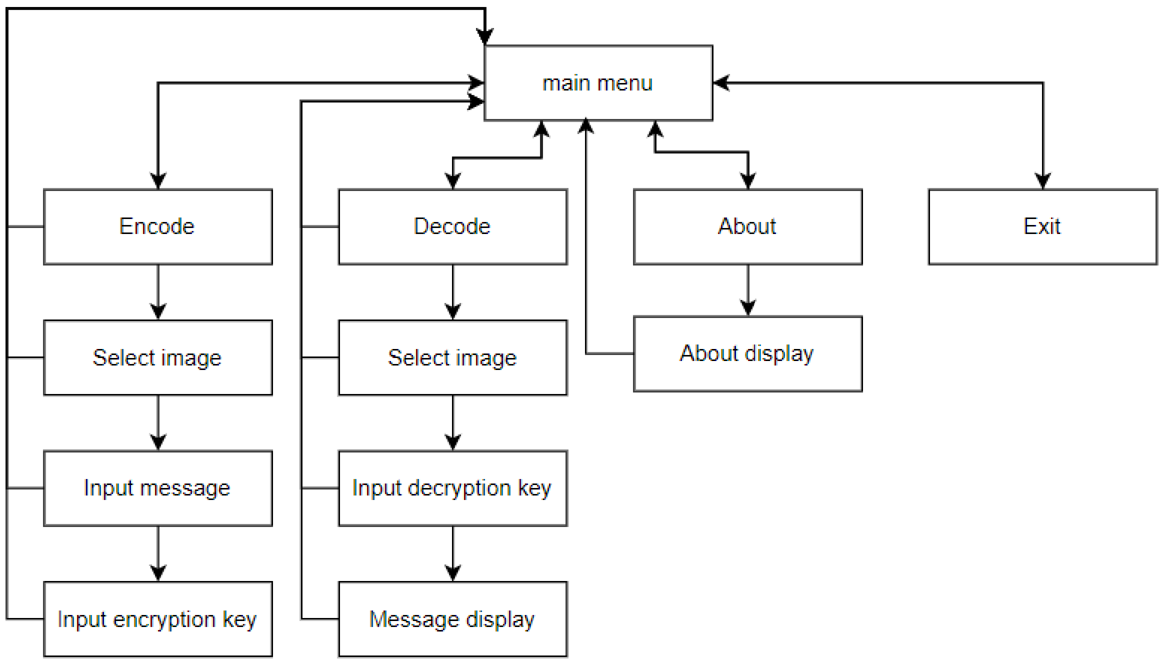


Figure 5: Interface Structure Diagram

3.3 Algorithm Details

For the image steganography tool, developed in the project I have used Red Channel Least Significant Bit Method for Steganography (R-Channel LSB Method). This algorithm is one of the most efficient and simple to implement. This algorithm uses the red color pixel data and modifies it to insert the user's data in it. There are many variations of the algorithm for image steganography, instead of just red pixel we can also take whole color channel of the image, but it will significantly change the image quality also will take more time to loop through all the color channel.

This technique works best when the file is longer than the message file and if image is grayscale. When applying LSB techniques to each byte of a 24-bit image, three bits can be encoded into each pixel.

```
Pixels: (00100111 11101001 11001000)
        (00100111 11001000 11101001)
        (11001000 00100111 11101001)
A: 01000001
Result: (00100110 11101001 11001000)
        (00100110 11001000 11101000)
        (11001000 00100111 11101001)
```

LSB insertion is easy to implement, at the same time, easily attacked. Today steganography is being incorporated into digital technology. The techniques have been used to create the watermarks that are in our nation's currency, as well as encode music information in the ever-popular mp3 music file. Copyrights can be included in files, and fingerprints can be used to identify the people who break copyright agreements. While Slight modifications in the color palette and simple image manipulations will destroy the entire hidden message.

The algorithm is modified to follow the CIA triad. This algorithm includes the escape sequence technique to successfully identify the messages and password in the image also Uses AES to encrypt with R-Channel LSB Method, for more security.

Algorithm to hide message in image:

Step 1: Start

```
Step 2: Take message, password and image from user
Step 3: Encrypt the message with the password
Step 4: Encode the encrypted message into binary form
Step 5: Hash the password and attach an escape sequence in the end
Step 6: Attach the password before encoded message
Step 7: While encoded data is not hidden
            For each pixel in image
                Get red color code
                Encode the color code into 8-bit binary
                Replace the 2 bit from LSB with 2 bit MSB of encoded data
                Update the pixel data in image
                If encoded data is hidden
                    Encoded data is hidden
Step 8: Stop
```

Algorithm to extract message from image

```
Step 1: Start
Step 2: Take the password and image with hidden message form users
Step 3: For each pixel in image
            Get red color code
            Encode the color code into 8-bit binary
            Append the two from LSB of encoded color code to a string
            If last 32 bit of string is password escape sequence
                Store password bits as string
                Segment the password in 7-bit each and convert to ASCII character
                If password does not match
                Go to step 1
        If last 48 bit of string is message escape sequence
            Break
Step 4: Segment the message in 7-bit each
Step 5: Convert the segmented bits to corresponding ASCII character to get cypher text
Step 6: Use password to decrypt the extracted cypher
Step 7: Display the plain text
Step 8: Stop
```

Chapter 4: Implementation and Testing

4.1 Implementation

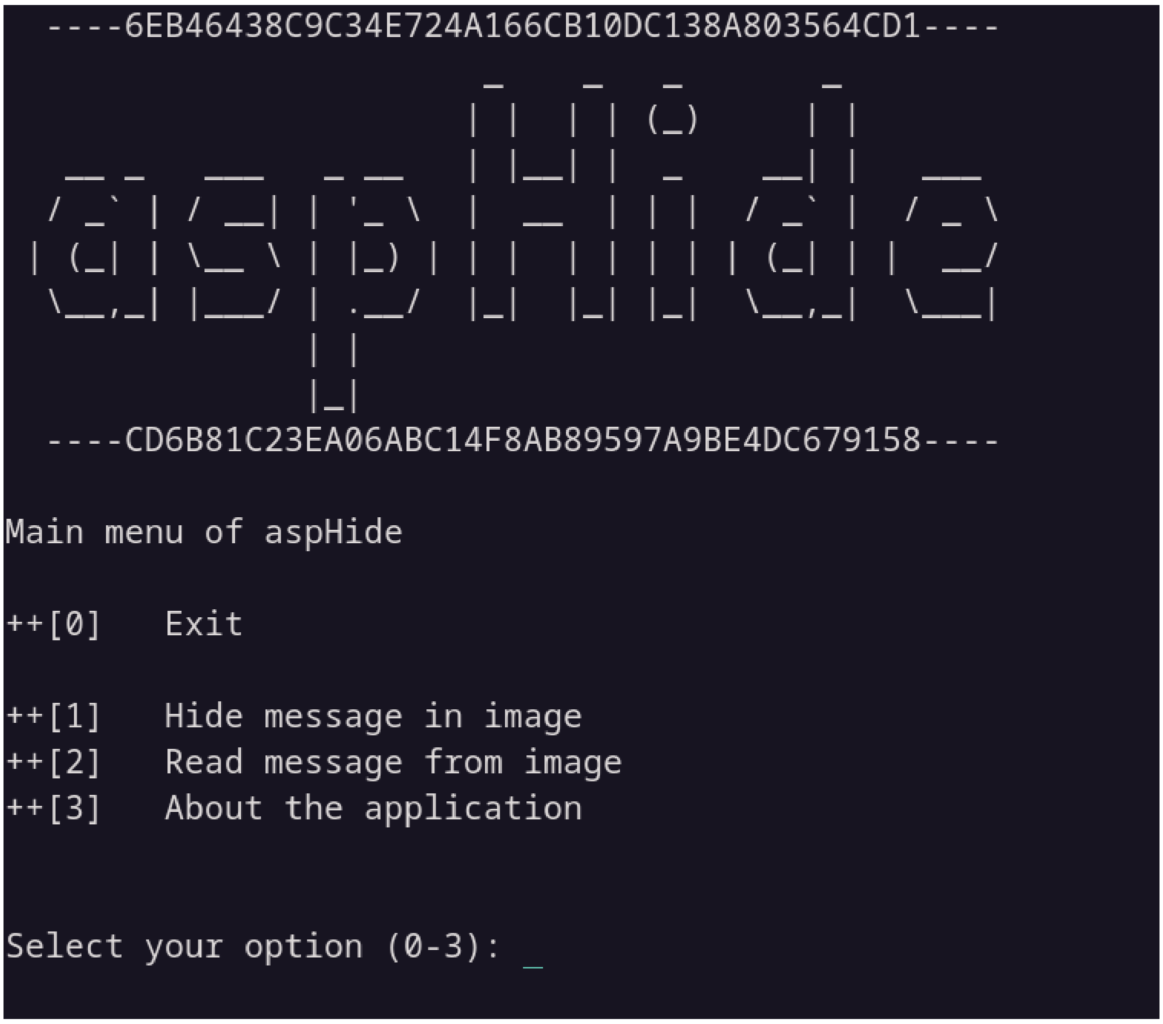
4.1.1 Tools Used

- **Python:** Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured, object-oriented and functional programming. [8] This project is fully powered by python and its various modules like Python Image Library (PIL), OS, time, cryptography, base64 and hashlib.
- **PyCharm**: PyCharm version 2023.2 is used as Integrated Development Environment for the project. All the development part of the project is done with the IDE.
- **Bash shell:** Bash is a Unix shell and command language written by Brian Fox for the GNU Project as a free software replacement for the Bourne shell. First released in 1989, it has been used as the default login shell for most Linux distributions. Bash was one of the first programs Linus Torvalds ported to Linux, alongside GCC. This tool is used to run the python file in terminal.
- **MS-Office:** The Microsoft office package is used in the various phases of the project.
- **Draw.io online:** This online tool is used to draw various required diagram during the project and documentation process. Draw.io is a free, online diagramming tool that allows you to create flowcharts, diagrams, mind maps, organisations charts, and much more. A web-based application, Draw.io is fully integrated with Google Drive.

4.1.2 Implementation details of modules

The project, aspHide is a modular application built by integrating various modules together. Here, multiple modules are developed and then integrated together to make the program to function as a whole.

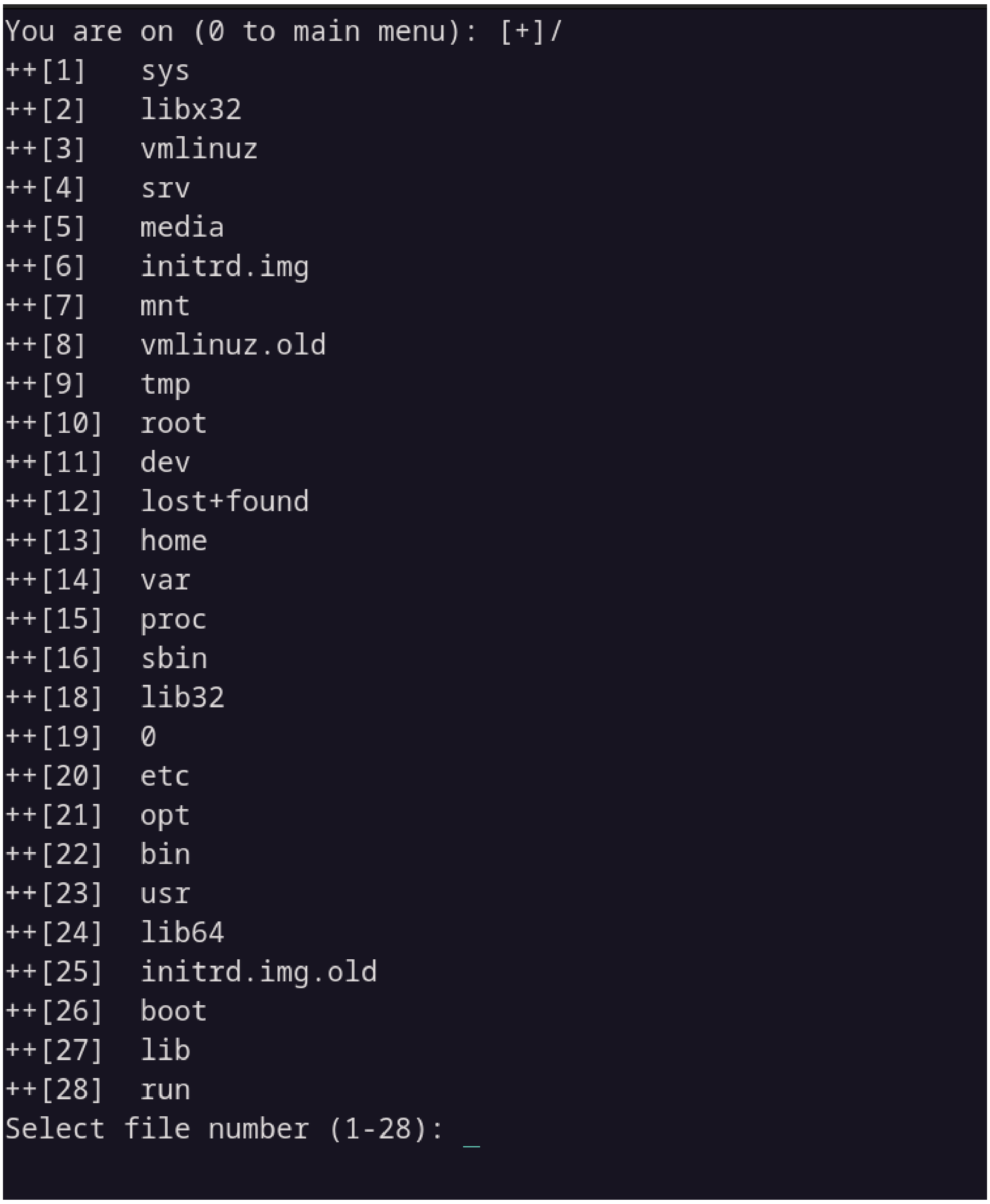
Menu Selector is a method in the program which allows users to traverse among the various options like, hiding message, viewing hidden message, exit, help etc. This module will read the user input and switch the program to do the task accordingly. This is also the index module or the first module to run once the program is initiated.



```
  ----6EB46438C9C34E724A166CB10DC138A803564CD1----
                        _    _   _       _
                       | |  | | (_)     | |
   __ _   ___   _ __   | |__| |  _    __| |   ___
  / _` | / __| | '_ \  |  __  | | |  / _` |  / _ \
 | (_| | \__ \ | |_) | | |  | | | | | (_| | |  __/
  \__,_| |___/ | .__/  |_|  |_| |_|  \__,_|  \___|
               | |
               |_|
  ----CD6B81C23EA06ABC14F8AB89597A9BE4DC679158----

Main menu of aspHide

++[0]   Exit

++[1]   Hide message in image
++[2]   Read message from image
++[3]   About the application


Select your option (0-3): _
```

Figure 6: Menu selector of aspHide

Image Selector in the program allows users to choose the desired image in which data is to be hidden. This module accomplishes the task by displaying the available files and folders of the current directory, then user gives the numeric value of the file to choose from. If user selects folder than the user is traversed to the directory, if file is chosen than the file is tested for image and if and only if the chosen file is image, program selects the chosen file. This module also checks for the invalid input and throws error in the display.



```
You are on (0 to main menu): [+]/
++[1]   sys
++[2]   libx32
++[3]   vmlinuz
++[4]   srv
++[5]   media
++[6]   initrd.img
++[7]   mnt
++[8]   vmlinuz.old
++[9]   tmp
++[10]  root
++[11]  dev
++[12]  lost+found
++[13]  home
++[14]  var
++[15]  proc
++[16]  sbin
++[18]  lib32
++[19]  0
++[20]  etc
++[21]  opt
++[22]  bin
++[23]  usr
++[24]  lib64
++[25]  initrd.img.old
++[26]  boot
++[27]  lib
++[28]  run
Select file number (1-28): _
```

Figure 7: Image selector of aspHide

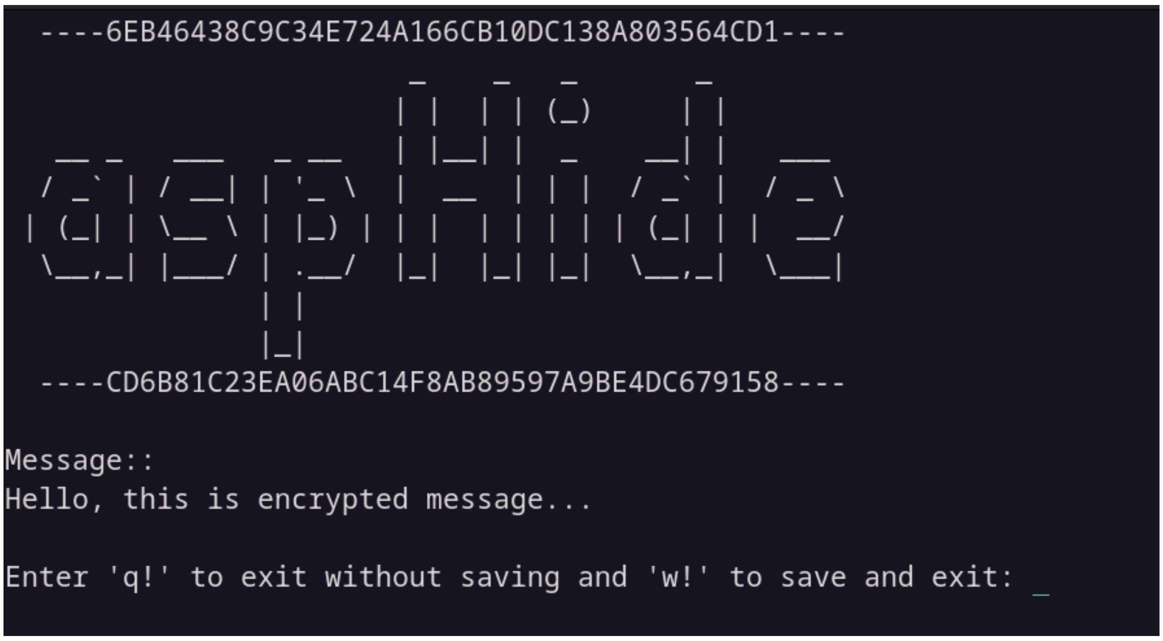
Message Encryption Module is responsible to get the encryption key and message from user, encrypt the message using AES algorithm with the key provided by user and return the cypher to the image generator module of the program. This module also hashes the password and attach escape sequence at the end of password hash and cypher text. This module makes the message unreadable to the hackers even if they get the image and are able to extract the pixel data.

Image Generator is a module that works along with the message encryption module and manipulates the pixel data of the image provided by users using image selector module. This module uses PIL to extract the pixel data of image before manipulating it. This module gets the image, updates the red pixel color code with the modified one, makes the copy of whole color pallet, generates a new image with hidden message in it and makes the copy of it in user's home directory.

Message Extractor Module is used if the user wants to see the hidden data of the user. This module extracts bits from the image and combines it to make a valid data, that can be processed to convert it into readable form. But before that user needs to provide the decryption key, if and only if the provided key is same as encryption key, the hidden cypher is extracted from image. Once the message is extracted, the escape sequences are trimmed from string and only genuine cypher is passed to message decryption module.

Message Decryption Module uses the data passed by the message extractor module and utilizes the provided key to decrypt the message successfully. This module uses Cryptography module of the python library.

Message Extractor is a module of the program that acts as a display for the extracted message and also is used as the message exporter, which saves the decrypted message from the program into a file in user's home directory. If user do not want to export the message than they can safely navigate to the main menu, using instructed commands.



```
  ----6EB46438C9C34E724A166CB10DC138A803564CD1----
                         _    _   _       _
                        | |  | | (_)     | |
   __ _   ___   _ __    | |__| |  _    __| |   ___
  / _` | / __| | '_ \   |  __  | | |  / _` |  / _ \
 | (_| | \__ \ | |_) |  | |  | | | | | (_| | |  __/
  \__,_| |___/ | .__/   |_|  |_| |_|  \__,_|  \___|
                | |
                |_|
  ----CD6B81C23EA06ABC14F8AB89597A9BE4DC679158----

Message::
Hello, this is encrypted message...

Enter 'q!' to exit without saving and 'w!' to save and exit: _
```

Figure 8: Message exporter module of aspHide

4.2 Testing

4.2.1 Test Cases for Unit Testing

Here, in the unit testing all the major functions being provided by each module of the project are tested with predefined conditions and outcomes are mentioned below.

Table 2: Test case for menu sector

Test case	Objective	Task	Expected Outcome	Obtained Outcome
Menu selection with invalid option	To select task from the menu	User inputs a random number	An error message is displayed	Failed
Provides option listed in menu	To select task from the menu	User inputs a number shown in the menu	Image selection option will be available	Passed

Table 3: Test case for image selector

Test case	Objective	Task	Expected Outcome	Obtained Outcome
Image selection using invalid path	To provide image to the program	User inputs a random path to image	An error message is displayed	Failed
Selecting non image files	To provide image to the program	User selects text file	An error message is displayed	Failed
Providing non existing index of the files	To provide image to the program	User provide an alpha numeric value	An error message is displayed	Failed
Providing valid input	To provide image to the program	Use provides available index value of file	User is asked to provide encryption key	Passed

Table 4: Test case for message decryption

Test case	Objective	Task	Expected Outcome	Obtained Outcome
Decrypting message with false password	To decrypt the encrypted message	User inputs a random password	User is directed to main menu with error	Failed
Decrypting message with true password	To decrypt encrypted password	User inputs correct password	User is provides with decrypted text	Passed

Table 5: Test case for message exporter

Test case	Objective	Task	Expected Outcome	Obtained Outcome
Saving message with random input	To export message in text file	User provides random input	An error message is displayed	Failed
Saving message with quit command	To export message in text file	User provides the quit command	User is directed to main menu	Failed
Saving message with save and quit command	To export message in text file	Use provides input for save and quit	The message file is saved and location is displayed	Passed

4.2.2 Test Case for System Testing

The program is fully tested for its performance by providing various inputs. The test results are shown below.

Table 6: System testing for aspHide

S.N	Test Case	Objectives	Expected Outcome	Remarks
1	Selecting menu options	To check whether the indexed function are accessible	The corresponding function to the index should run	Passed
2	Selecting image	To check if only images are being selected	Only files with JPG, PNG and JPEG should be selected	Passed
3	Generating new image	To check is new image with message is generated in user's home directory	An image, looking same as the processed image should be in user's home directory	Passed
4	Decrypting message	To check if the message is in human readable form	The displayed message should be in human readable form	Passed
5	Exporting message	To check if the decrypted message is saved in text file	The message should be saved in user's home directory	Passed
6	Accessing menu from other modules	To check if user can navigate to menu while using other features	User should be redirected to menu, discarding the current task	Passed
7	Exiting the program	To check if user can exit program without causing error	User should get exit message and exit the program	Passed
8	Setting up system	To check if alias for program is made in bashrc file	User must be able to access program from any directory using its name	Passed

Chapter 5: Conclusion and Future Recommendations

5.1 Conclusion

In this transition phase of technology, this program will help people to share their confidential data among the other users in a safe manner. The organizations or the individuals who cannot afford to create a secure tunnel like GRE and cannot use IPSec like technology can definitely use this program to hide their confidential data while sharing over public internet.

As always, the program is only as secure as the user wants it to be. There may some consideration and precautions that user may have to follow to gain the optimum of the program. The encryption key length, its complexity will give the strength to the encryption. The system is designed and implemented to have the basic features that a steganography tool must have. In future the tool will be updated and upgraded to meet with the user requirements.

5.2 Lesson Learnt

It has been a matter of immense pleasure, honor and challenge to have this opportunity to take up this project and complete it successfully. The project is designed to meet the current requirements that was visible to me as a necessity to be fulfilled. This led to an intensive research and reviews of existing systems, which changed my way of studying and researching.

- I have learnt to analyze a problem from user's point of view and how to make it user-friendly by hiding its complexity behind the user interface.
- Project allowed me to understand and analyze my self-skill and interest that will definitely be helpful for me, moving further.
- I learnt the way of studying the necessity and requirement of the environment and convert it to a product that solves the necessity.
- Improved the secure coding skill and developed the habit of making the code simple to read.
- Habit of dissecting the problem and solving it in a modular manner is developed.

5.3 Future Recommendation

Following points are recommended for the betterment of the project in long run:

- User interface will be updated by analyzing the user feedback.

- The encryption technology will be updated if a better algorithm is developed.
- The compatibility issue for command prompt will be solved.
- Protection for the exported messages will provided in future updates.

REFERENCES

[1] Identity Digital, "named.com," Named, [Online]. Available: https://www.name.com/support/articles/360007888013-banned-country-list-rapidssl-geotrust-symantec-ssls. [Accessed 15 10 2023].

[2] M. Semilof, "techtarget.com," 08 2023. [Online]. Available: https://www.techtarget.com/searchsecurity/definition/steganography#:~:text=Steganography%20software%20is%20used%20to,the%20data%20to%20be%20hidden.. [Accessed 26 10 2023].

[3] V. L. Reddy, "ieeexplore.ieee.org," 15 04 2011. [Online]. Available: https://ieeexplore.ieee.org/document/6172093. [Accessed 26 10 2023].

[4] infosecinstitute, "resources.infosecinstitute.com," [Online]. Available: https://resources.infosecinstitute.com/topics/cryptography/steganography-and-tools-to-perform-steganography/. [Accessed 26 10 2023].

[5] infosecinstitute, "resources.infosecinstitute.com," infosecinstitute, [Online]. Available: https://resources.infosecinstitute.com/topics/cryptography/steganography-and-tools-to-perform-steganography/. [Accessed 26 10 2024].

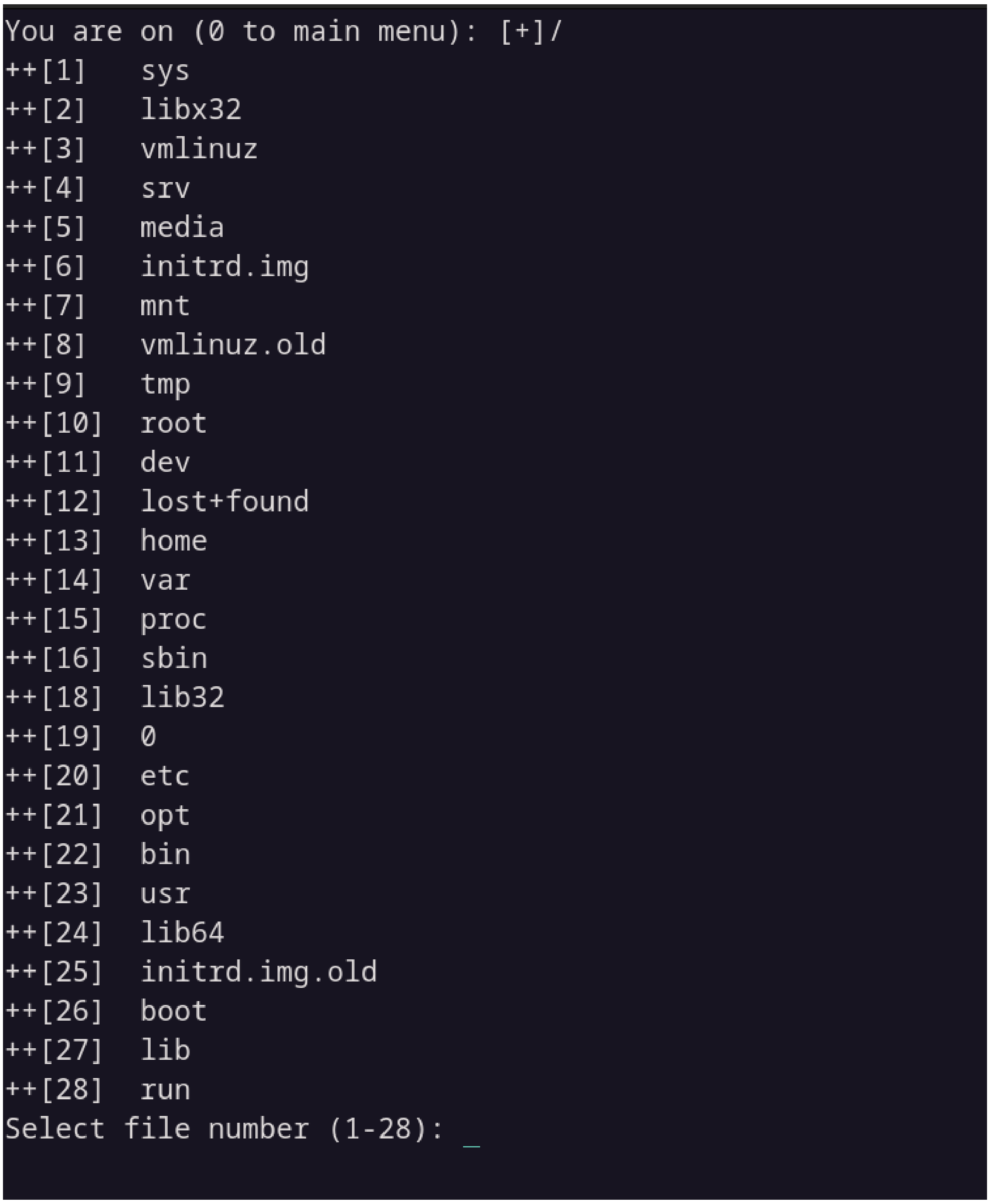
[6] syvaidya, "github.com," 8 12 2008. [Online]. Available: https://github.com/syvaidya/openstego. [Accessed 26 10 2023].

[7] A. S. A. V. Avni Aggarwal, "ripublication.com," 2019. [Online]. Available: https://www.ripublication.com/irph/ijisaspl2019/ijisav11n1spl_19.pdf. [Accessed 26 10 2024].

[8] Python Software Foundation, "en.wikipedia.org," 23 09 2023. [Online]. Available: https://en.wikipedia.org/wiki/Python_(programming_language). [Accessed 31 10 2023].

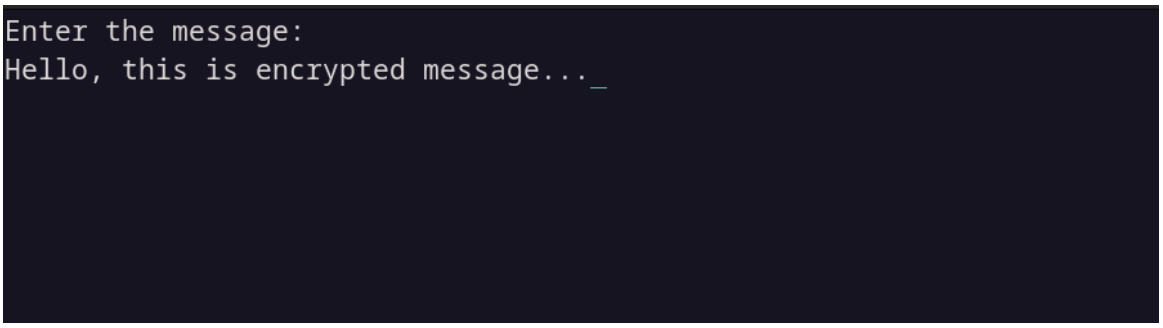
Appendices

Screenshots:



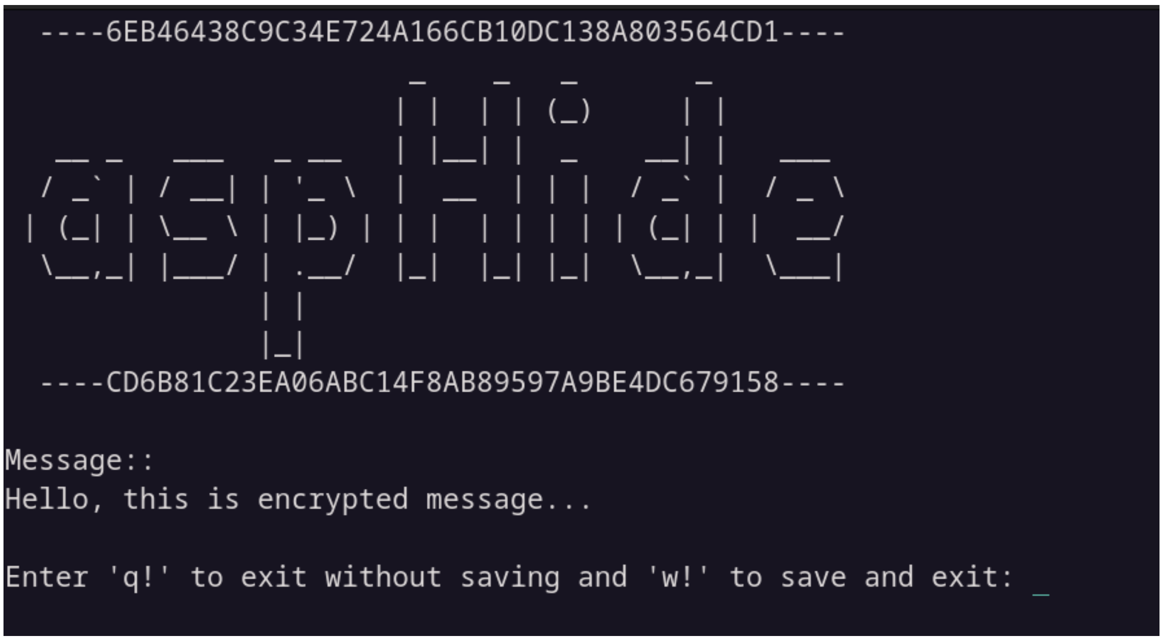
```
You are on (0 to main menu): [+]/
++[1]   sys
++[2]   libx32
++[3]   vmlinuz
++[4]   srv
++[5]   media
++[6]   initrd.img
++[7]   mnt
++[8]   vmlinuz.old
++[9]   tmp
++[10]  root
++[11]  dev
++[12]  lost+found
++[13]  home
++[14]  var
++[15]  proc
++[16]  sbin
++[18]  lib32
++[19]  0
++[20]  etc
++[21]  opt
++[22]  bin
++[23]  usr
++[24]  lib64
++[25]  initrd.img.old
++[26]  boot
++[27]  lib
++[28]  run
Select file number (1-28): _
```

Figure 9: Image selector



```
Enter the message:
Hello, this is encrypted message...
```

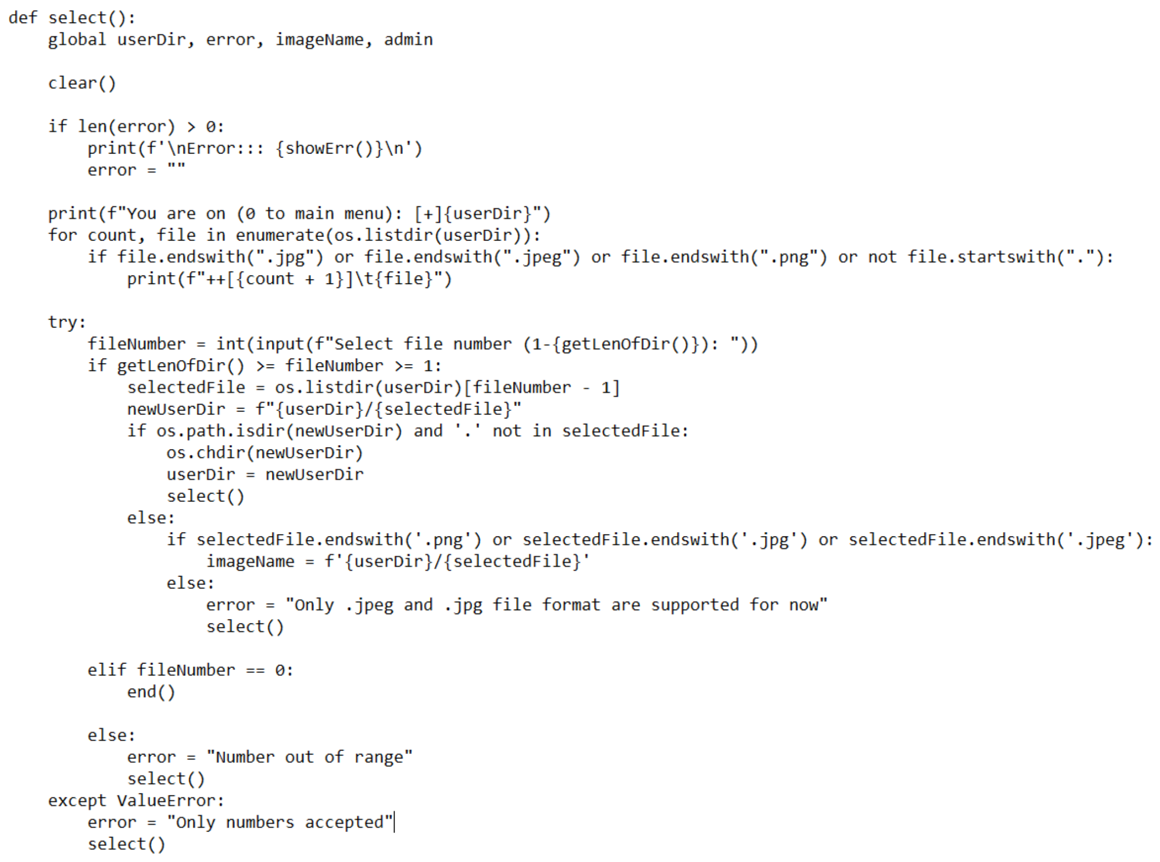
Figure 10: Message input field



```
  ----6EB46438C9C34E724A166CB10DC138A803564CD1----
                        _    _   _       _
                       | |  | | (_)     | |
   __ _   ___   _ __   | |__| |  _    __| |   ___
  / _` | / __| | '_ \  |  __  | | |  / _` |  / _ \
 | (_| | \__ \ | |_) | | |  | | | | | (_| | |  __/
  \__,_| |___/ | .__/  |_|  |_| |_|  \__,_|  \___|
               | |
               |_|
  ----CD6B81C23EA06ABC14F8AB89597A9BE4DC679158----

Message::
Hello, this is encrypted message...

Enter 'q!' to exit without saving and 'w!' to save and exit:
```

Figure 11: Message display

Source code for image selection:



```
def select():
    global userDir, error, imageName, admin

    clear()

    if len(error) > 0:
        print(f'\nError::: {showErr()}\n')
        error = ""

    print(f"You are on (0 to main menu): [+]{userDir}")
    for count, file in enumerate(os.listdir(userDir)):
        if file.endswith(".jpg") or file.endswith(".jpeg") or file.endswith(".png") or not file.startswith("."):
            print(f"++[{count + 1}]\t{file}")

    try:
        fileNumber = int(input(f"Select file number (1-{getLenOfDir()}): "))
        if getLenOfDir() >= fileNumber >= 1:
            selectedFile = os.listdir(userDir)[fileNumber - 1]
            newUserDir = f"{userDir}/{selectedFile}"
            if os.path.isdir(newUserDir) and '.' not in selectedFile:
                os.chdir(newUserDir)
                userDir = newUserDir
                select()
            else:
                if selectedFile.endswith('.png') or selectedFile.endswith('.jpg') or selectedFile.endswith('.jpeg'):
                    imageName = f'{userDir}/{selectedFile}'
                else:
                    error = "Only .jpeg and .jpg file format are supported for now"
                    select()

        elif fileNumber == 0:
            end()

        else:
            error = "Number out of range"
            select()
    except ValueError:
        error = "Only numbers accepted"
        select()
```

Figure 12: Source code of image selection