LSB IMAGE STEGANOGRAPHY

Done by

S.YASHVANTH

OBJECTIVES

- To decode hidden data embedded inside an image using LSB (Least Significant Bit) steganography.
- To extract and verify a predefined magic string that confirms the presence of encoded secret data.
- To retrieve the secret file's metadata, including its extension and size, from the encoded image.
- To reconstruct the original secret file by decoding its contents bit-by-bit from the image.
- To save the extracted data into an output file, restoring the original hidden message.

FOUNDATIONAL REQUIREMENTS

• C Programming Fundamentals → Variables, pointers, dynamic memory allocation, bitwise operations.

• String Functions → strcmp, strcpy, strlen for validating magic string, file extensions, and decoded data.

• File Handling (Binary Mode) → fopen, fread, fwrite, fseek for reading encoded image data and writing output files.

• Modular Programming → Separate functions for decoding magic string, extension, size, and secret data to maintain clean design.

• Bit Manipulation Logic → Extracting LSB bits from each byte of the image to reconstruct characters and file contents.

FEATURES

- Magic String Verification → Confirms the presence of hidden data by validating a predefined magic signature.
- File Extension Extraction → Decodes and reconstructs the original secret file extension from image LSBs.
- Secret File Size Retrieval → Extracts the encoded size information to allocate accurate memory for secret data.
- LSB Bit Decoding → Reads the least significant bits of image bytes to rebuild characters and binary content.
- Dynamic Memory Utilization → Adjusts buffer sizes using malloc/realloc for flexible decoding of variable-length data.
- Output File Reconstruction → Writes the decoded secret content into a newly generated file matching the original format.
- Error Handling & Validation → Detects invalid magic strings, incorrect file sizes, missing data, and corrupted encodings.

PROGRAM FLOW

Command-line Interface: ./a.out -d <stego_image.bmp> <output_file>

File Validation: Checks if the input image exists and is in BMP format.

Confirms successful opening of the encoded image file.

Header Processing: Skips the 54-byte BMP header reserved for metadata.

Ensures the image is large enough to contain encoded data.

Magic String Verification: Reads the LSBs of initial bytes to reconstruct the magic string.

Validates the magic signature to confirm hidden data presence.

Data Decoding: Sets error_flag for invalid tokens or formatting.

Stops token generation if errors are found.

Output File Generation: Creates a new file using the decoded extension.

Writes the reconstructed secret data into the output file.

```c
#include <stdio.h>
#include "encode.h"
#include "types.h"
#include <string.h>
#include "common.h"

Status do_encoding(EncodeInfo *encInfo)
{
    if(open_files(encInfo)==e_success)
    {
        printf("Opening Files Done...\n");
        if(check_capacity(encInfo)==e_success)
        {
            printf("Checking the capacity done...\n");
            if(copy_bmp_header(encInfo->fptr_src_image, encInfo->fptr_stego_image) ==  e_success)
            {
                printf("Header Copied Successfully...\n");
                if(encode_magic_string(MAGIC_STRING,encInfo) == e_success)
                {
                    printf("Encoded Magic string Successfully...\n");
                    if(encode_secret_file_extn_size(encInfo->extn_secret_file,encInfo)==e_success)
                    {
                        printf("Encoded secret File extention Size Successfully...\n");
                        if(encode_secret_file_extn(encInfo->extn_secret_file,encInfo)==e_success)
                        {
                            printf("Encoded secret File extention Successfully...\n");
                            if(encode_secret_file_size(encInfo->size_secret_file,encInfo)==e_success)
                            {
                                printf("Encoded secret File Data Successfully...\n");
                                if(encode_secret_file_data(encInfo)==e_success)
                                {
                                    if(copy_remaining_img_data(encInfo->fptr_src_image,encInfo->fptr_stego_image)== e_success)
                                    {
                                        printf("Copied Remaining Data...\n");
                                        if(close_all_files(encInfo)== e_success)
                                        {
                                            printf("Files Closing Done...\n");
                                        }
```

beautiful.bmp | encode.c | encode.h | secret.txt | common.h | main.c | types.h | decode.c | output.txt | stego.bmp

STEGANO_25021F > encode.h > ...

```c
#ifndef ENCODE_H
#define ENCODE_H
#include <string.h>
#include "common.h"
#include "types.h" // Contains user defined types
/*
 * Structure to store information required for
 * encoding secret file to source Image
 * Info about output and in termediate data is
 * also stored
 */
#define MAX_SECRET_BUF_SIZE 1
#define MAX_IMAGE_BUF_SIZE (MAX_SECRET_BUF_SIZE * 8)
#define MAX_FILE_SUFFIX 4

typedef struct _EncodeInfo
{
    /* Source Image info */
    char *src_image_fname;
    FILE *fptr_src_image;
    uint image_capacity;
    uint bits_per_pixel;
    char image_data[MAX_IMAGE_BUF_SIZE];

    /* Secret File Info */
    char *secret_fname;
    FILE *fptr_secret;
    char extn_secret_file[MAX_FILE_SUFFIX];
    char secret_data[MAX_SECRET_BUF_SIZE];
    long size_secret_file;

    /* Stego Image Info */
    char *stego_image_fname;
    FILE *fptr_stego_image;

} EncodeInfo;

/* Encoding function prototype */
```

File Edit Selection View Go Run Terminal Help

Emertxe

beautiful.bmp | encode.c | encode.h | secret.txt | common.h | main.c | types.h | decode.c | output.txt | stego.bmp

STEGANO_25021F > secret.txt

```
1  Hala Madrid
```

File Edit Selection View Go Run Terminal Help

Emertxe

beautiful.bmp | encode.c | encode.h | secret.txt | common.h | main.c | types.h | decode.c | output.txt | stego.bmp

STEGANO_25021F > common.h > ...

```
1  #ifndef COMMON_H
2  #define COMMON_H
3
4  /* Magic string to identify whether stegged or not */
5  #define MAGIC_STRING "#*"
6
7  #endif
8
```

```c
#include <stdio.h>
#include <string.h>
#include "encode.h"
#include "types.h"

int main(int argc,char *argv[])
{
    EncodeInfo encInfo;
    DecodeInfo decInfo;
    uint img_size;
    if (argc < 3)
    {
        printf("Enter the Arguments ! \n");
        printf("For Encoding args : ./a.out -e filename.bmp secret.txt\n");
        printf("For Decoding args : ./a.out -d encoded_image.bmp\n");
        return 1;
    }
    else
    {
        OperationType res = check_operation_type(argv);
        if(res == e_encode)
        {

            if(argc < 3)
            {
                printf("Enter the Arguments ! \n");
                printf("For Encoding args : ./a.out -e filename.bmp secret.txt\n");
                return e_failure;
            }
            printf("Encoding Operation Enabled...\n");
            Status ret = read_and_validate_encode_args(argv,&encInfo);
            if(ret==e_success)
            {
                printf("Arguments Verified SuccessFully...\n");
                do_encoding(&encInfo);
            }
        }
        else if(res==e_decode)
```

File Edit Selection View Go Run Terminal Help

Emertxe

beautiful.bmp | encode.c | encode.h | secret.txt | common.h | main.c | types.h | decode.c | output.txt | stego.bmp

STEGANO_25021F > types.h > __unnamed_enum_08bf_1

```c
#ifndef TYPES_H
#define TYPES_H

/* User defined types */
typedef unsigned int uint;

/* Status will be used in fn. return type */
typedef enum
{
    e_success,
    e_failure
} Status;

typedef enum
{
    e_encode,
    e_decode,
    e_unsupported
} OperationType;

#endif

```

```c
#include <stdio.h>
#include "encode.h"
#include "types.h"
#include <string.h>
#include <stdlib.h>
#include "common.h"

static unsigned char ch;

Status read_and_validate_decode_args(char *argv[], DecodeInfo *decInfo)
{
    if(strstr(argv[2],".bmp"))
    {
        decInfo->decode_image_fname = argv[2];
        decInfo->fptr_decode_image = fopen(decInfo->decode_image_fname,"r");
        if(decInfo->fptr_decode_image == NULL)
        {
            printf("Opening the file Failed\n");
            return e_failure;
        }
        if(argv[3] == NULL)
        {
            strcpy(decInfo->output_secret_file_fname,"output");
            return e_success;
        }
        else
        {
            strcpy(decInfo->output_secret_file_fname, strtok(argv[3],"."));
            return e_success;
        }
    }
    else
    {
        return e_failure;
    }
}

Status do_decoding(DecodeInfo *decInfo)
```

beautiful.bmp | encode.c | encode.h | secret.txt | common.h | main.c | types.h | decode.c | output.txt | stego.bmp

STEGANO_25021F > output.txt

```
1   Hala Madrid
```

File Edit Selection View Go Run Terminal Help

Emertxe

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

wsl

```
yashvanth@YASH:/mnt/c/Emertxe/STEGANO_25021F$ gcc *.c
yashvanth@YASH:/mnt/c/Emertxe/STEGANO_25021F$ ./a.out -e beautiful.bmp secret.txt
Encoding Operation Enabled...
Arguments Verified SuccessFully...
Opening Files Done...
width = 1024
height = 768
Size of Secret File : 11
Checking the capacity done...
Header Copied Successfully...
Encoded Magic string Successfully...
Encoded secret File extention Size Successfully...
Encoded secret File extention Successfully...
Encoded secret File Data Successfully...
Copied Remaining Data...
Files Closing Done...
yashvanth@YASH:/mnt/c/Emertxe/STEGANO_25021F$
```

wsl

File Edit Selection View Go Run Terminal Help

Emertxe

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

wsl

```
yashvanth@YASH:/mnt/c/Emertxe/STEGANO_25021F$ gcc *.c
yashvanth@YASH:/mnt/c/Emertxe/STEGANO_25021F$ ./a.out -e beautiful.bmp secret.txt
Encoding Operation Enabled...
Arguments Verified SuccessFully...
Opening Files Done...
width = 1024
height = 768
Size of Secret File : 11
Checking the capacity done...
Header Copied Successfully...
Encoded Magic string Successfully...
Encoded secret File extention Size Successfully...
Encoded secret File extention Successfully...
Encoded secret File Data Successfully...
Copied Remaining Data...
Files Closing Done...
yashvanth@YASH:/mnt/c/Emertxe/STEGANO_25021F$ ./a.out -d stego.bmp
Decoding Operation Enabled...
Arguments Verified SuccessFully...
Magic String Verified Sucessfully...
Decoded Magic string Successfully...
Decoded Secret File Extension Size Successfully...
Decoded Secret File Extension Successfully...
Decoded Secret File Size Successfully...
Decoded Secret File Data Successfully...
Closing All the Files...
Decoding Done Successfully...
yashvanth@YASH:/mnt/c/Emertxe/STEGANO_25021F$
```

wsl

File Edit Selection View Go Run Terminal Help

Emertxe

beautiful.bmp | encode.c | encode.h | secret.txt | common.h | main.c | types.h | decode.c | output.txt | stego.bmp

STEGANO_25021F > stego.bmp

CORE TAKEAWAYS

• Deep Understanding of Steganography Design: Gained hands-on experience with Least Significant Bit (LSB)–based data hiding and retrieval from BMP image files.

• Strong Mastery of C Programming Fundamentals: Enhanced skills in pointers, arrays, file operations, memory allocation, and bitwise manipulation.

• Practical Implementation of Decoding Logic: Designed and developed functions to extract magic strings, file extensions, file sizes, and secret data from encoded images.

• Advanced Bit-Level Data Processing Skills: Learned to reconstruct characters by combining LSB-extracted bits, ensuring accurate conversion from binary patterns to readable data.