Secure and Dependable Systems
Jacobs University Bremen
Dr. Jürgen Schönwälder

Course: CO21-320203
Date: 2019-05-10
Due: 2019-05-17

SADS 2019 Problem Sheet #6

Problem 6.1: *steganograhy – pnmhide* (6+2+2 = 10 points)

Your task is to write a program `pnmhide` that can hide text in an image file. We will provide you with a template program that takes care of reading and writing image files and which implements the command line interface. You have to implement only a function `im_encode()` that encodes a character string in an image and a function `im_decode()` that decodes a character string from an image. The program makes use of the `libnetpbm` library for reading and writing images using the PBM, PGM, and PPM image formats.

The usage of pnmhide is best explained using an example. Lets assume we have a covertext image in `cover.pnm`. We can create a stegotext image in `stego.pnm` as follows:

```
$ # Hide the text "hello world" in the stegotext image stego.pnm
$ pnmhide -e "hello world" cover.pnm > stego.pnm
$ # Retrieve the hidden text from the stegotext image stego.pnm
$ pnmhide -d stego.pnm
hello world
$ # Detect if there is no hidden text in an image (do not produce garbage)
$ pnmhide -d cover.pnm
```

The work can be broken down into the following steps:

a) Implement `im_encode()` and `im_decode()` functions.

b) Reliably detect the presence of hidden text.

c) Define and implement a strategy to minimize the visual impact.

```c
/*
 * pnmhide/src/image.h --
 */

#ifndef _IMAGE_H
#define _IMAGE_H

typedef struct image {
    xel **xels;          /*!< Two-dimensional array of pixels */
    xelval maxval;       /*!< The maximum value of a pixel */
    int cols;            /*!< The number of columns */
    int rows;            /*!< The number of rows */
    int format;          /*!< The external format of the image */
} image_t;

/**
 * \brief Read an image from a file. If the file name is NULL, read
 *        from the standard input.
 *
 * \param im pointer to the image.
 * \param name of the image file.
 * \result 0 on success, positive error code on failure
 */

int
im_read(image_t *im, char *file);

/**
 * \brief Write an image to a file. If the file name is NULL, write
 *        to the standard input.
 *
 * \param im pointer to the image.
 * \param name of the image file.
 * \result 0 on success, positive error code on failure
 */

int
im_write(image_t *im, char *file);

/**
 * \brief Free the memory allocated for an image.
 *
 * \param im pointer to the image.
 */

void
im_free(image_t *im);

#endif
```

```c
/*
 * pnmhide/src/hide.h --
 */

#ifndef _HIDE_H
#define _HIDE_H

#include "image.h"

/**
 * \brief Encode a message in an image.
 *
 * \param im pointer to the image.
 * \param msg message to encode.
 * \result length of the message encoded or negative number on error.
 */

int
im_encode(image_t *im, char *msg);

/**
 * \brief Decode a message from an image.
 *
 * \param im pointer to the image.
 * \param buffer buffer to fill with the decoded message.
 * \result length of the message decoded or negative number on error.
 */

int
im_decode(image_t *im, char *buffer, size_t size);

#endif
```