Seguridad Informática: Informe de prácticas

Práctica 1. Esteganografía

Carlos González Recio – 70885474A

10 de mayo de 2017

Introducción

La esteganografía (del griego *στεγανος* (*steganos*):cubierto u oculto, y *γραφος* (*graphos*): escritura), trata el estudio y aplicación de técnicas que permiten ocultar mensajes u objetos, dentro de otros, llamados portadores**,** de modo que no se perciba su existencia. Es decir, procura ocultar mensajes dentro de otros objetos y de esta forma establecer un canal encubierto de comunicación, de modo que el propio acto de la comunicación pase inadvertido para observadores que tienen acceso a ese canal.

Una forma de diferenciar la esteganografía con la criptografía común es que la criptografía solo cifra los archivos manteniendo el archivo original visible pero al abrirlo mostrara una secuencia de caracteres que no permitirá su lectura y para ver su contenido original es necesario conocer la clave. En la esteganografía pude verse un archivo con un formato diferente y para conocer su contenido original será necesario conocer la clave y el software con el que se ocultó.

Existen numerosos métodos y algoritmos utilizados para ocultar la información dentro de archivos multimedia, pero la que se va a realizar en esta práctica es la siguiente

Inserción en el bit menos significativo (LSB)

Este es el método moderno más común y popular usado para esteganografía, también es uno de los llamados métodos de sustitución. Consiste en hacer uso del bit menos significativo de los píxeles de una imagen y alterarlo. La misma técnica puede aplicarse a vídeo y audio, aunque no es lo más común. Hecho así, la distorsión de la imagen en general se mantiene al mínimo (la perceptibilidad es prácticamente nula), mientras que el mensaje es esparcido a lo largo de sus píxeles. Esta técnica funciona mejor cuando el archivo de imagen es grande, posee fuertes variaciones de color ("imagen ruidosa") y también aventaja cuanto mayor sea la profundidad de color. En general, los mejores resultados se obtienen en imágenes con formato de color RGB (tres bytes, componentes de color, por píxel).

Ejemplo práctico

El valor (1 1 1 1 1 1 1 1) es un número binario de 8 bits. Al bit ubicado más a la derecha se le llama "bit menos significativo" (LSB) porque es el de menor peso, alterándolo cambia en la menor medida posible el valor total del número representado.

Un ejemplo de esteganografía: Ocultamiento de la letra "A". Si se tiene parte de una imagen con píxeles con formato RGB (3 bytes), su representación original podría ser la siguiente (3 píxeles, 9 bytes):

(1 1 0 1 1 0 1 0) (0 1 0 0 1 0 0 1) (0 1 0 0 0 0 1 1)

(0 0 0 1 1 1 1 0) (0 1 0 1 1 0 1 1) (1 1 0 1 1 1 1 1)

(0 0 0 0 1 1 1 0) (0 1 0 0 0 1 1 1) (0 0 0 0 0 1 1 1)

El mensaje a cifrar es ‘A’ cuya representación ASCII es (1 0 0 1 0 1 1 1), entonces los nuevos píxeles alterados serían:

(1 1 0 1 1 0 1 **1**) (0 1 0 0 1 0 0 **0**) (0 1 0 0 0 0 1 **0**)

(0 0 0 1 1 1 1 **1**) (0 1 0 1 1 0 1 **0**) (1 1 0 1 1 1 1 **1**)

(0 0 0 0 1 1 1 **1**) (0 1 0 0 0 1 1 **1**) (0 0 0 0 0 1 1 1)

Observar que se ha sustituido el bit del mensaje (letra A, marcados en negritas) en cada uno de los bits menos significativos de color de los 3 píxeles. Fueron necesarios 8 bytes para el cambio, uno por cada bit de la letra A, el noveno byte de color no se utilizó, pero es parte del tercer pixel (su tercera componente de color).

El método del LSB funciona mejor en los archivos de imágenes que tienen una alta resolución y usan gran cantidad de colores. En caso de archivos de audio, favorecen aquellos que tienen muchos y diferentes sonidos que poseen una alta tasa de bits.

Desarrollo

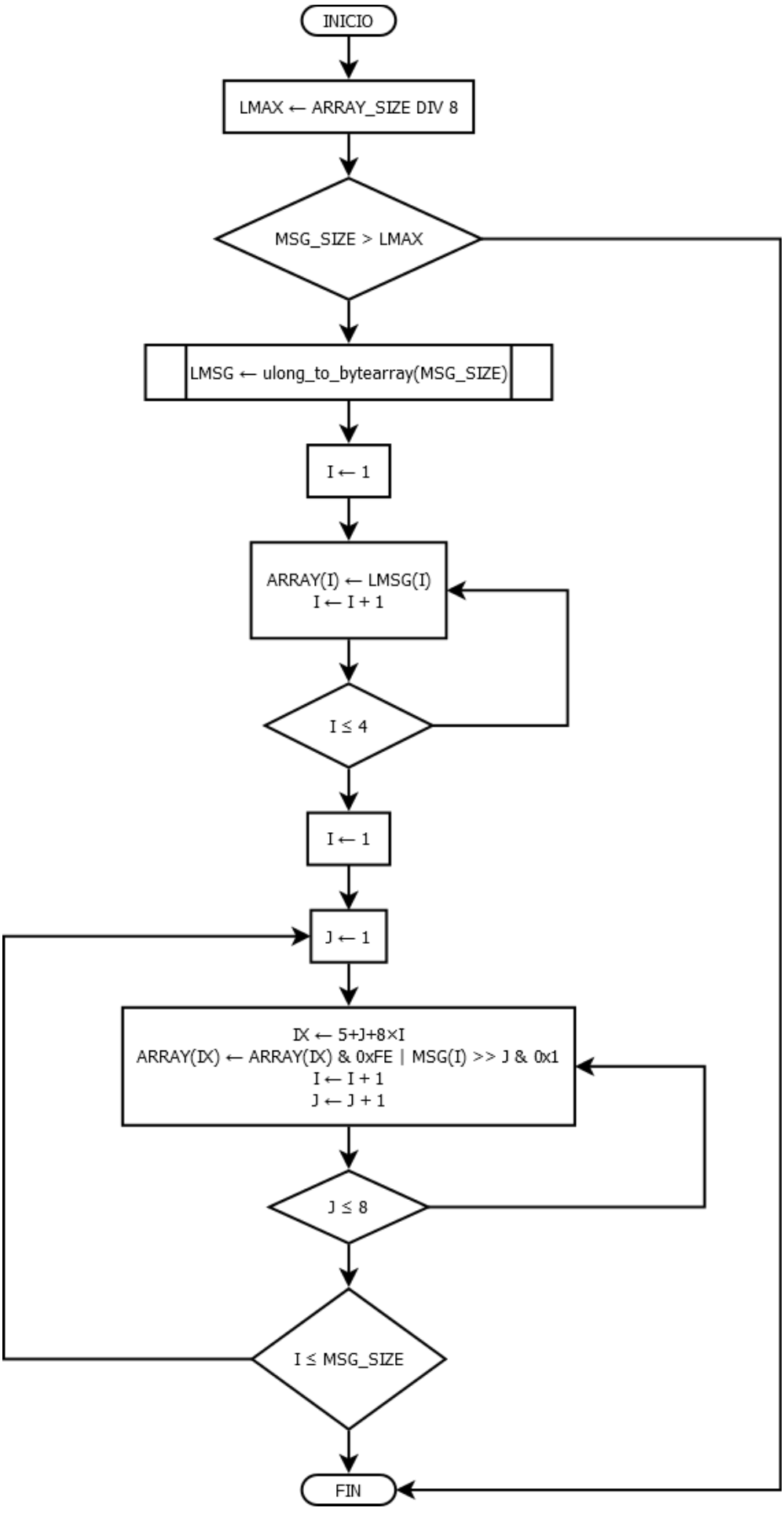
Se ha escogido el lenguaje de programación C para la realización de la práctica (determinado por uno de los requisitos de la práctica), y se utilizan tanto las librerías estándar como algunas funciones del sistema (necesarias para el tratamiento de ficheros).

El programa se compone de varias funciones auxiliares:

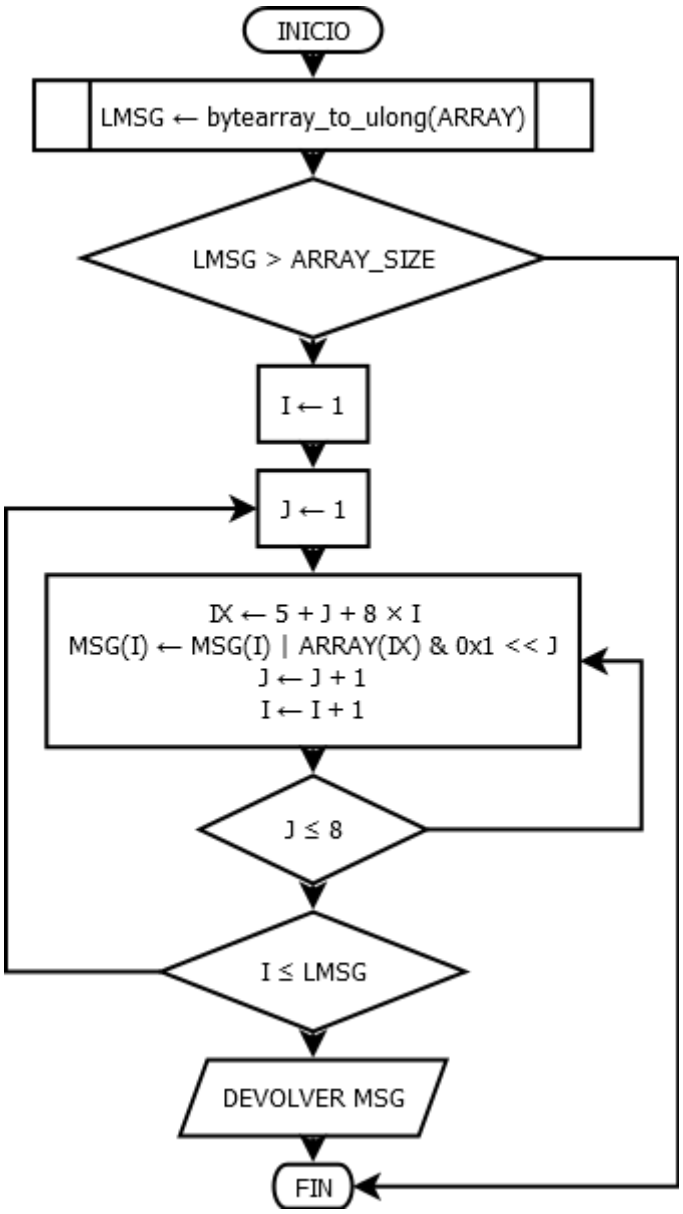
- `void uso(char *ruta)`
  Imprime el uso del ejecutable.
- `FILE *abrir(char *ruta, char *modo)`
  Abre el fichero en la `ruta` especificada con el `modo` de lectura y escritura dado.
- `void cerrar(FILE *fp)`
  Cierra el fichero designado por el puntero de fichero `fp`.
- `long fsize(FILE *fp)`
  Devuelve el tamaño del fichero al que apunta `fp`.
- `void *mapear(FILE *fp, size_t tam, int prot, int flags)`
  Función de envoltura sobre la llamada a la función `mmap`.
- `unsigned char *ulong_to_bytearray(unsigned long num)`
  Convierte un entero largo sin signo a un array de caracteres.
- `unsigned long bytearray_to_ulong(unsigned char *array)`
  Realiza la conversión inversa de la función anterior.

A continuación, se mostrará el diagrama de flujo de las funciones principales:

- `void encode(unsigned char **array, size_t array_size, unsigned char *mensaje, size_t msg_size)`
  Realiza el método de codificación por el bit menos significativo, guardando en `array`, de tamaño `array_size`, el `mensaje` (de longitud `msg_size`) con el siguiente procedimiento:

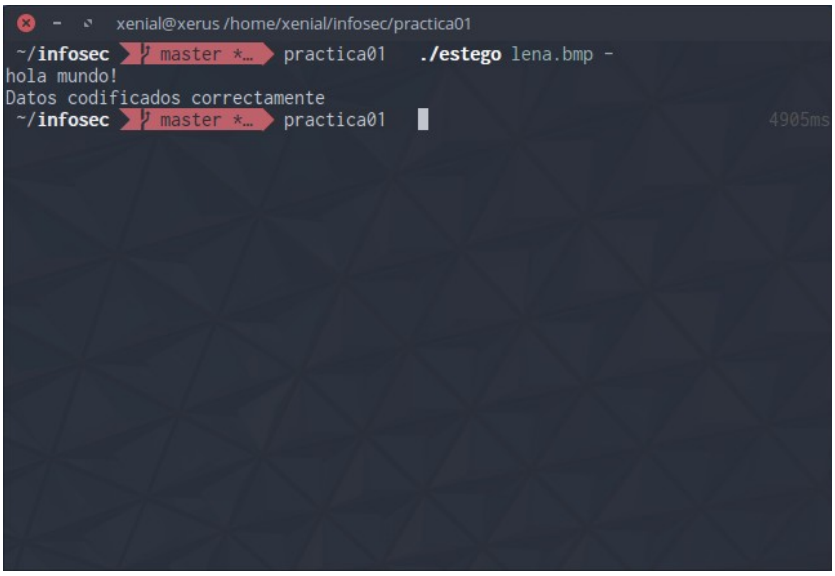

INICIO
LMAX ← ARRAY_SIZE DIV 8
MSG_SIZE > LMAX
LMSG ← ulong_to_bytearray(MSG_SIZE)
I ← 1
ARRAY(I) ← LMSG(I)
I ← I + 1
I ≤ 4
I ← 1
J ← 1
IX ← 5+J+8×I
ARRAY(IX) ← ARRAY(IX) & 0xFE | MSG(I) >> J & 0x1
I ← I + 1
J ← J + 1
J ≤ 8
I ≤ MSG_SIZE
FIN

- `unsigned char *decode(unsigned char *array, size_t array_size)`
  Decodifica el mensaje guardado en `array` (de tamaño `array_size`) realizando el proceso inverso a la función anterior:

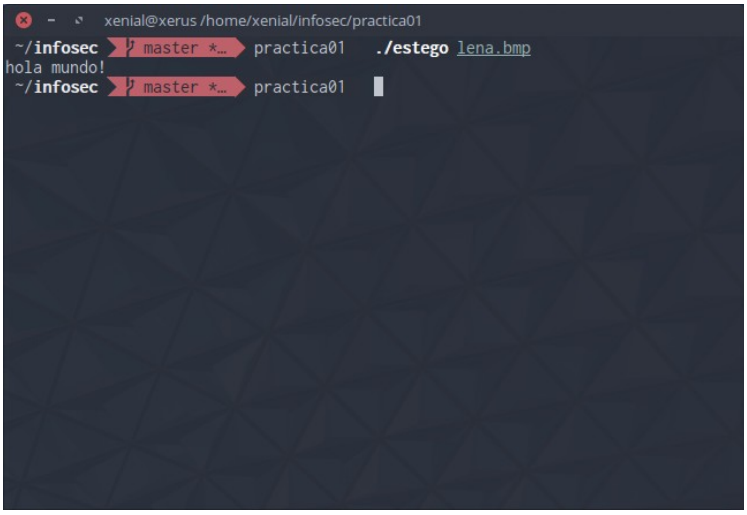


Pruebas

Imagen de referencia: Lenna (sin datos ocultos)



Codificación del mensaje “hola mundo!” en la imagen.



Extracción del mensaje de la imagen.

Referencias

- https://es.wikipedia.org/wiki/Esteganografía