



INSTITUTO TECNOLÓGICO DE BUENOS AIRES

Trabajo Práctico Especial

Deyheralde, Ben

Mutz, Matías

Quian Blanco, Francisco

Stanfield, Theo

Criptografía y Seguridad - 72.44

Segundo cuatrimestre 2024 - Grupo 5

Tabla de contenidos

Introducción.....	3
Cuestiones a analizar.....	4
I. Discutir los siguientes aspectos relativos al documento.....	4
a. Organización formal del documento.....	4
b. La descripción del algoritmo.....	4
c. La notación utilizada, ¿es clara? ¿Hay algún error o contradicción?.....	5
II. Esteganografiar un mismo archivo en un .bmp con cada uno de los tres algoritmos, y comparar los resultados obtenidos. Hacer un cuadro comparativo de los tres algoritmos estableciendo ventajas y desventajas.....	5
III. Explicar detalladamente el procedimiento realizado para descubrir qué se había ocultado en cada archivo y de qué modo. Indicar qué se encontró en cada archivo.....	6
IV. Algunos mensajes ocultos tenían, a su vez, otros mensajes ocultos. Indica cuál era ese mensaje y cómo se había ocultado.....	10
V. Uno de los archivos ocultos era una porción de un video de una película, donde se ve ejemplificado una manera de ocultar información ¿qué se ocultaba y sobre qué portador?.	10
VI. ¿De qué se trató el método de esteganografiado que no era LSB1 ni LSB4 ni LSBI? ¿Es un método eficaz? ¿Por qué?.....	11
VII. ¿Por qué la propuesta del documento de Majeed y Sulaiman es realmente una mejora respecto de LSB común?.....	11
VIII. En la implementación se optó por guardar los patrones invertidos antes del mensaje ¿de qué otra manera o en qué otro lugar podría guardarse el registro de los patrones invertidos?.....	12
IX. ¿Qué dificultades encontraron en la implementación del algoritmo del paper?.....	12
X. ¿Qué mejoras o futuras extensiones harías al programa stegobmp?.....	13

Introducción

La esteganografía es una técnica de ocultación de mensajes que permite insertar información dentro de un archivo portador de manera que su presencia pase desapercibida para observadores no autorizados. A diferencia de la criptografía, que hace ilegible un mensaje pero no oculta su existencia, la esteganografía disimula la presencia misma del mensaje, lo cual lo convierte en una opción muy poderosa para proteger información sensible en el contexto digital.

Este proyecto implementa un programa en lenguaje C para aplicar técnicas de esteganografía en imágenes con formato BMP como portadores del mensaje encubierto. Se desarrollaron y probaron algoritmos de esteganografiado de bits menos significativos (LSB), específicamente LSB1, LSB4 y LSBI, que permiten modificar distintos niveles de profundidad en el píxel para ocultar información de manera eficiente. Además, para ofrecer una capa adicional de seguridad, el programa incluye la opción de cifrar el mensaje mediante algoritmos criptográficos antes de su inserción en el portador, asegurando que, incluso si se detectara la existencia de datos ocultos, el contenido permanece protegido.

Este informe detalla la implementación, los desafíos técnicos y las pruebas realizadas para garantizar que la información incrustada permanezca indetectable y segura dentro de la imagen portadora.

Cuestiones a analizar

I. Discutir los siguientes aspectos relativos al documento

a. Organización formal del documento.

Se considera que el documento sigue una estructura formal y bien organizada, típica de un artículo académico. La organización general del documento “An Improved LSB Image Steganography Technique Using Bit-Inverse In 24 Bit Colour Image” incluye las siguientes secciones:

- i. **Introducción:** Proporciona un contexto y una motivación clara para el estudio de este algoritmo, describiendo el problema que se aborda y la relevancia del mismo.
- ii. **Revisión de la Literatura:** Resume trabajos previos relevantes y posiciona el trabajo actual en el contexto de la investigación existente.
- iii. **Descripción del Algoritmo:** Detalla el algoritmo LSBI que se propone, explicando su funcionamiento y las mejoras que introduce sobre métodos existentes.
- iv. **Resultados Experimentales:** Presenta los resultados obtenidos mediante la aplicación del algoritmo propuesto y compara estos resultados con los de otros métodos comunes de esteganografía.
- v. **Conclusiones:** Resume lo que se logró con este estudio y sugiere posibles direcciones para futuras investigaciones.

b. La descripción del algoritmo.

El algoritmo LSBI se presenta como una mejora del método estándar LSB. La técnica propuesta incluye los siguientes pasos clave:

- i. Se elige una imagen de color de 24 bits, RGB, que servirá como portadora del mensaje secreto.
- ii. El mensaje secreto es una secuencia de bits que se oculta en la imagen portadora.
- iii. En lugar de simplemente reemplazar los LSBs (“Least significant bits”) de algunos píxeles de la imagen original con los bits del mensaje secreto, el algoritmo LSBI implementa una inversión de bits con los LSBs de los bytes seleccionados. La inversión se realiza tomando el segundo y tercer bit de cada byte a modificar, y según la cantidad de modificaciones para los patrones 00, 01, 10 y 11, se aplica inversión o no, de manera que la cantidad de bytes modificados disminuye de la original.
- iv. El algoritmo modifica píxeles en los bytes que corresponden a las componentes verde y azul de cada píxeles, mientras que los bytes correspondientes al color rojo se

saltean, para que la imagen original se modifique menos y de esta manera se agregue más seguridad y complejidad.

- v. La imagen se modifica para incluir el mensaje secreto.

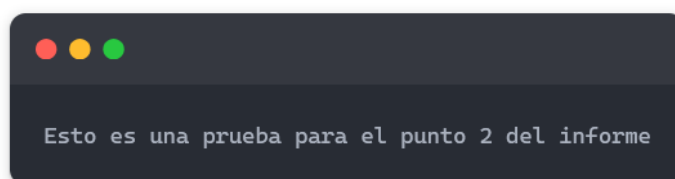
c. La notación utilizada, ¿es clara? ¿hay algún error o contradicción?

La notación utilizada en el documento es en general coherente y clara. No se han encontrado errores significativos en el artículo, aunque se ha identificado un pequeño error tipográfico en el siguiente párrafo: *“Calculate the pattern occurrences of these two bits on the cover-image, which are either 00, 10, 10, or 11. Classify the cover image according to these four patterns.”*

En este caso, en lugar de "00, 10, 10, or 11" debería decir "00, 01, 10, or 11". Este pequeño error no afecta significativamente la comprensión del documento, ya que es claro el contexto en el que se utiliza.

II. Esteganografiar un mismo archivo en un .bmp con cada uno de los tres algoritmos, y comparar los resultados obtenidos. Hacer un cuadro comparativo de los tres algoritmos estableciendo ventajas y desventajas.

Realizamos el esteganografiado de los distintos algoritmos ocultando el siguiente mensaje:



El resultado del mismo con los tres algoritmos fue el siguiente:

LSB1	LSB4	LSBI

Como se puede observar, no resulta perceptible a simple vista los cambios realizados en el procedimiento, por más que la imagen utilizada haya sido de baja resolución por utilizar un contenido de archivo corto.

Podemos comparar conceptualmente los tres algoritmos, notando ciertas ventajas y desventajas en cada uno de ellos:

Algoritmo	Ventajas	Desventajas
LSB1	<ul style="list-style-type: none"> ➤ Fácilmente implementable. ➤ Cambios poco perceptibles (al modificar únicamente el bit menos significativo de cada componente). 	<ul style="list-style-type: none"> ➤ Ocupa mucho espacio, ya que cada bit del mensaje se almacena en un byte del portador, por lo que se necesitan 8 veces el tamaño del mensaje como espacio para almacenar. ➤ Fácil de identificar.
LSB4	<ul style="list-style-type: none"> ➤ Fácilmente implementable. ➤ Ocupa menos espacio que LSB1 porque cada 4 bits del mensaje se almacenan en un byte del portador, entonces en vez de requerir 8 veces el tamaño, solo se necesita el doble. 	<ul style="list-style-type: none"> ➤ Más fácil de identificar que LSB1 porque la mitad de cada byte de la imagen se ve modificada. ➤ El impacto de la modificación es mayor por lo que es más perceptible.
LSBI	<ul style="list-style-type: none"> ➤ Muy bajo impacto visual, porque no solo se modifica el último bit, sino que solo aplica a componentes azules y verdes, y utilizando el patrón de inversión, los bytes realmente modificados son muy pocos. 	<ul style="list-style-type: none"> ➤ Mayor complejidad de implementación. ➤ El almacenamiento requerido es aún mayor que LSB1 porque no solo no se aprovecha la componente roja de cada pixel, sino que también hay que almacenar el patrón

III. Explicar detalladamente el procedimiento realizado para descubrir qué se había ocultado en cada archivo y de qué modo. Indicar qué se encontró en cada archivo.

Lo primero que se hizo para descubrir el contenido oculto en cada archivo, se creó un script en bash para probar todos los archivos mediante “fuerza bruta” utilizando todos los métodos de esteganografía. Esto se realizó para identificar cualquier indicio que indicara qué archivo estaba relacionado con cada método. A continuación el script utilizado:

```
#!/bin/sh

cd "$(dirname "$0")"

files="$(find grupo5 -type f -name '*.bmp')"
stegs='LSB1 LSB4 LSBI'

rm -rf extracts
[ -d extracts ] || mkdir extracts

for file in $files; do
    for steg in $stegs; do
        tail=$(basename $file)
        extract="extracts/${tail%.*}_${steg}"
        printf "\nTrying to extract $file with $steg...\n"
        bin/stegobmp --extract -p $file --out $extract --steg $steg 1>/dev/null
        if [ $? -eq 0 ]; then
            printf "Extracted $file with $steg\n"
            break
        else
            printf "Failed to extract $file with $steg\n"
        fi
    done
done
printf "\n===== \n"
```

Los resultados que se obtuvieron de haber corrido este script fueron los siguientes:

```
Trying to extract grupo5/titanic.bmp with LSB1 ...
Failed to extract grupo5/titanic.bmp with LSB1

Trying to extract grupo5/titanic.bmp with LSB4 ...
Failed to extract grupo5/titanic.bmp with LSB4

Trying to extract grupo5/titanic.bmp with LSBI ...
Failed to extract grupo5/titanic.bmp with LSBI

=====

Trying to extract grupo5/hugo4.bmp with LSB1 ...
Failed to extract grupo5/hugo4.bmp with LSB1

Trying to extract grupo5/hugo4.bmp with LSB4 ...
Extracted grupo5/hugo4.bmp with LSB4

=====

Trying to extract grupo5/roma.bmp with LSB1 ...
Failed to extract grupo5/roma.bmp with LSB1

Trying to extract grupo5/roma.bmp with LSB4 ...
Failed to extract grupo5/roma.bmp with LSB4

Trying to extract grupo5/roma.bmp with LSBI ...
Extracted grupo5/roma.bmp with LSBI

=====

Trying to extract grupo5/quito.bmp with LSB1 ...
Failed to extract grupo5/quito.bmp with LSB1

Trying to extract grupo5/quito.bmp with LSB4 ...
Failed to extract grupo5/quito.bmp with LSB4

Trying to extract grupo5/quito.bmp with LSBI ...
Failed to extract grupo5/quito.bmp with LSBI
```

El script nos dio además de dos archivos (“hugo4_LSB4.png” y “roma_LSB1.pdf”), información clave para que podamos avanzar y obtener más información. Los dos archivos obtenidos fueron al extraer los archivos “hugo.bmp” y “roma.bmp” con los algoritmos “LSB4” y “LSBI” respectivamente. El contenido de la imagen es el siguiente buscaminas:



Del archivo “roma.bmp” utilizando el algoritmo LSBI se obtuvo el archivo PDF antes mencionado que tiene como información “*al.png cambiarle la extensión por .zip y descomprimir*”.

De los otros dos archivos obtuvimos la información de que “titanic.bmp” no fue esteganografiado con

alguno de los métodos LSB sin utilizar una contraseña y de que “quito.bmp”, se debe extraer con el algoritmo LSB1 pero se cree que el contenido fue encriptado anteriormente al esteganografiado y con la resolución del buscaminas tal vez se consiga lo que falta para obtener su información.

Al realizar lo que se pedía en el PDF con el script que se muestra a continuación se logró entender lo que debíamos hacer con el buscaminas.

```
cp extracts/hugo4_LSB4.{png,zip}
unzip extracts/hugo4_LSB4.zip -d extracts
```

Se obtuvo la siguiente información:

```
cada mina es un 1.
cada fila forma una letra.
Los ascii de las letras empiezan todos en 01.
Así encontraras el algoritmo que tiene clave de 128 bits y el modo
La password esta en otro archivo
Con algoritmo, modo y password hay un .wmv encriptado y oculto.
```

Es aquí donde el buscaminas fue resuelto, obteniendo el siguiente tablero con los resultados:

1	1	1	1	3	🚩
🚩	3	2	🚩	5	🚩
🚩	🚩	2	3	🚩	🚩
3	3	1	3	🚩	🚩
🚩	2		3	🚩	5
🚩	2		2	🚩	🚩

Ahora, cambiando los números y espacios en blanco por 0s y las banderas por 1s y agregando los caracteres 01 al principio de cada fila se obtienen los 8 bits correspondientes a letras individuales que formarán una palabra que indica el algoritmo de cifrado de clave de 128 bits y el modo de bloque a utilizar:

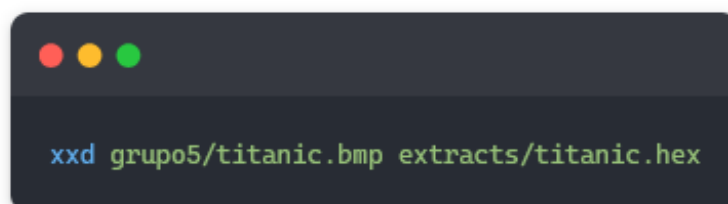


Como se puede ver, el mensaje formado es “AesCbc”, es decir, nos indica que el contenido del archivo .wmv, se encuentra encriptado utilizando el algoritmo AES128 con modo de bloque CBC.

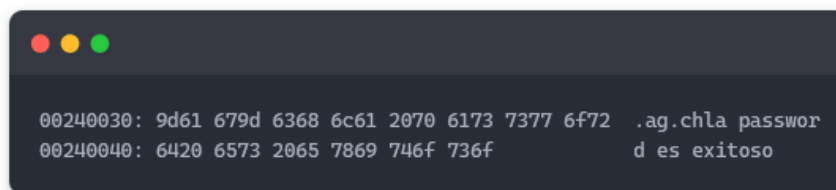
Con esta información, lo único que queda por descubrir para descryptar el contenido, es la contraseña del mismo. Se indicó en el archivo descomprimido que la contraseña estaría presente en otro archivo. Sabemos por consigna que 3 archivos tienen información esteganografiada y un cuarto archivo tiene contenido oculto por otro método. Por el análisis del des-esteganografiado por fuerza bruta que se realizó antes, ya se estima que el archivo .bmp con el contenido del .wmv es el archivo “quito.bmp” debido al gran tamaño del archivo. Al ser un archivo muy grande contiene una capacidad para guardar mensajes muy grandes, como por ejemplo un video (.wmv). Por lo tanto, queda libre el archivo “titanic.bmp” que debe contener la contraseña.

Originalmente, explorando la imagen, a primera vista salta la palabra “TITANIC”, lo que hizo suponer la posibilidad de que la contraseña fuera dicho texto. Probando con distintas combinaciones de mayúsculas y minúsculas, y viendo el fallo en cada intento, se descartó totalmente la idea.

Leyendo nuevamente la consigna del trabajo práctico, notamos que se declara lo siguiente: “Se recomienda, para este análisis, usar algún editor de archivos binarios (hex editor neo o similar) que permita hacer comparaciones de los archivos”. Por esta razón, se decidió analizar dicho archivo con herramientas de visualización de contenido a nivel bytes. Para esto se decidió utilizar la herramienta por línea de comandos llamada xxd.



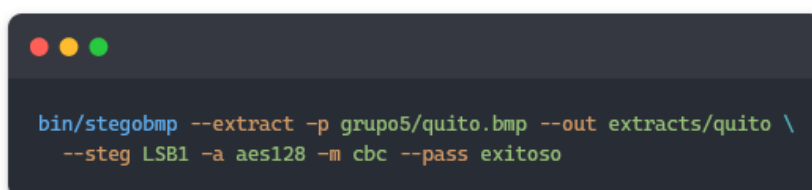
Al abrir el archivo y explorando su contenido, al final del mismo se puede observar un mensaje oculto en los últimos bytes del mismo.



```
00240030: 9d61 679d 6368 6c61 2070 6173 7377 6f72  .ag.chla passwor
00240040: 6420 6573 2065 7869 746f 736f                d es exitoso
```

“La password es exitoso”

Obteniendo ya la contraseña (“exitoso”) se puede correr el comando final, especificando los parámetros para descryptar el contenido averiguados previamente:



```
bin/stegobmp --extract -p grupo5/quito.bmp --out extracts/quito \
--steg LSB1 -a aes128 -m cbc --pass exitoso
```

Una vez ejecutado dicho comando, se obtiene el archivo .wmv esperado, concluyendo la búsqueda del tesoro. El archivo resultante forma parte de un fragmento de la película *Wanted*.

IV. Algunos mensajes ocultos tenían, a su vez, otros mensajes ocultos. Indica cuál era ese mensaje y cómo se había ocultado.

Durante el proceso de análisis del esteganografiado, un archivo ya oculto contenía otro contenido también oculto. Este archivo es el que nombramos “hugo4_LSB4.png”. Al obtener el otro archivo (en formato PDF), el mismo nos informaba que el archivo con extensión .png también contenía un formato válido como comprimido .zip. Es decir, al renombrar el archivo de “hugo4_LSB4.png” a “hugo4_LSB4.zip” y descomprimirlo utilizando alguna utilidad de descompresión que soporte el formato ZIP, se podía obtener un contenido adicional, es decir, una carpeta llamada “sols” junto con un archivo en formato .txt llamado “sol5.txt”, el cual contenía instrucciones específicas relacionadas a la imagen .png obtenida antes de cambiar la extensión del archivo.

Para ocultar el comprimido ZIP dentro de la imagen PNG muy posiblemente se realizó la técnica de *archivo políglota*. La misma consiste en la generación de archivos que puedan ser interpretados por diferentes programas. En el caso de los archivos PNG, el

contenido de la imagen se guarda en bloques conocidos como *chunks* y se marca la finalización del contenido de la imagen con un bloque conocido como IEND. Los archivos PNG permiten guardar información más allá de dicho bloque manteniendo una estructura válida para programas que lo interpreten. Es decir, los visualizadores de imagen PNG leen cada chunk de datos hasta encontrar el bloque IEND donde finalizan la lectura, más allá de dicho bloque simplemente lo ignoran.

En el caso de los archivos ZIP, el mismo contiene un encabezado con datos relacionados al archivo, indicando el inicio del contenido. Los lectores de archivos ZIP, reciben el archivo que se desea descomprimir y buscan entre los bytes del mismo hasta encontrar dicho encabezado, es decir, que todo lo previo al encabezado es ignorado.

De esta manera, seguramente los bytes del archivo ZIP fueron agregados luego del chunk IEND del archivo PNG, formando así un archivo polígloa, que al abrirlo con un visualizador de PNG, se puede reconocer la imagen porque lee hasta el chunk de IEND, y, al descomprimir el mismo archivo con algoritmo ZIP, el programa busca el encabezado del comprimido (luego del chunk IEND), procesa el contenido y lo descomprime, ignorando lo que se encuentra antes de dicho chunk.

V. Uno de los archivos ocultos era una porción de un video de una película, donde se ve ejemplificado una manera de ocultar información ¿qué se ocultaba y sobre qué portador?

El método de ocultamiento que se utilizó en el video corresponde a ocultar un mensaje dentro de un tejido. A la hora de tejer algunos hilos, estos se salteaban la trama por encima de otros. En caso de que el hilo vertical este encima por encima de los demás corresponde un 1, y en caso de que sea el hilo horizontal corresponde un 0. De esta manera se podía ocultar código binario, y por ende ocultar cualquier tipo de texto. En el caso del fragmento de la película, se ocultaban nombres de personas.

VI. ¿De qué se trató el método de esteganografiado que no era LSB1 ni LSB4 ni LSBI? ¿Es un método eficaz? ¿Por qué?

El método de esteganografiado que no era ninguno el LSB1, LSB4 ni LSBI era un método muy simple que modifica los bytes de una imagen para colocar el mensaje secreto. Este fue el utilizado para esconder la contraseña dentro del archivo “titanic.bmp”, el mismo

consistió en modificar los últimos bytes del archivo colocando la representación en hexadecimal del mensaje *“La password es exitoso”*.

Se considera que este método no es para nada eficaz ya que utilizando una herramienta para ver el contenido de esos bytes, como en el caso de nuestro grupo que utilizó la herramienta xxd, es muy fácil observar el mensaje oculto.

Sabiendo que hay un mensaje oculto es muy intuitivo intentar obtener el mensaje de esta manera, esta es otra de las razones por las que creemos que no es eficaz.

VII. ¿Por qué la propuesta del documento de Majeed y Sulaiman es realmente una mejora respecto de LSB común?

La mejora que han propuesto Majeed y Sulaiman es una mejora significativa a los métodos LSB1 y LSB4 principalmente porque al saltarse las componentes rojas de cada pixel se incrementa el nivel de ocultamiento del mensaje. Esto sucede porque el ojo humano es sensible al color rojo y una modificación en el mismo haría muy evidente dichos cambios, lo que perjudica al ocultamiento efectivo. Además, esto hace que la imagen original haya sido modificada en menor escala, y por tanto, es más difícil llegar a darse cuenta de que dentro de esa imagen hay un mensaje oculto.

La inversión de los bits propuesta permite reducir efectivamente la cantidad de bytes que cambiaron respecto de la imagen original, haciendo menos perceptible todavía el ocultamiento del mensaje.

Teniendo en cuenta todos estos puntos, se logra que LSBI sea un algoritmo que brinda mayor seguridad y por tanto resulte una mejora importante con respecto a LSB1 y LSB4.

VIII. En la implementación se optó por guardar los patrones invertidos antes del mensaje ¿de qué otra manera o en qué otro lugar podría guardarse el registro de los patrones invertidos?

Los patrones invertidos podrían haber sido guardados de distintas maneras aparte de ser guardados antes del mensaje. Algunas opciones que pensamos fueron, incluir estos patrones en el encabezado del archivo de modo que no se rompa la integridad del archivo y

de esta manera no ser visibles a simple vista. También podrían ser ocultadas al final del mensaje, para que, al revisar los bits del archivo no se detecte ninguna anomalía (ya que LSBI cambia pocos bits) hasta llegar al final del archivo donde se encuentran estos patrones. De esta manera se genera más complejidad para detectar que haya algo oculto. Otra opción podría ser guardar los patrones en un solo byte en vez de guardarlos en bytes separados, y de esta manera disminuir la longitud del archivo. Por último, otra manera que se nos ocurrió para ocultar estos patrones, es insertarlos al principio del archivo pero de distinta manera. Esto implica que en vez de utilizar los primeros cuatro bytes del archivo, se podría saltar la componente roja de los píxeles al igual que el algoritmo del LSBI, o tal vez saltarse dos componentes para incrementar la dificultad de detección incluso más.

IX. ¿Qué dificultades encontraron en la implementación del algoritmo del paper?

La mayor dificultad que encontramos a la hora de implementar el algoritmo LSBI fue comprender el ejemplo dado en el artículo para poder replicarlo. Al principio pensamos en utilizar las funciones del algoritmo LSB1 que ya habíamos implementado, pero tras prueba y error entendimos que frente a varios de los cambios proporcionados por el artículo no iba a ser factible reutilizar estas funcionalidades. Estos cambios presentados, como por ejemplo omitirse las componentes rojas, comparar los bits del archivo portador con el archivo esteganografiado, calcular qué patrones debían ser invertidos y cuáles no, guardarse estos patrones, no nos resultaron difíciles de implementar luego de haber podido comprender los cambios adecuadamente. Tuvimos una inconveniencia menor al implementar el algoritmo que surgió de no saltarse las componentes rojas, pero una vez solucionado este problema pudo funcionar correctamente.

X. ¿Qué mejoras o futuras extensiones harías al programa **stegobmp**?

Unas posibles mejoras o futuras extensiones que se nos ocurrieron para el programa **stegobmp** fueron las siguientes. Incluir otros métodos tanto como para esteganografiado, como también para encriptado. También se podría agregar la posibilidad de poder firmar el mensaje previamente a ser encriptado. De esta manera, utilizando criptografía asimétrica, nos aseguramos que el emisor del mensaje sea el esperado. A su vez se debería poder encriptar con una clave pública (previo al esteganografiado), para asegurarnos de que el receptor del mensaje sea el adecuado. También sería una buena idea poder implementar

esteganografiado en otros tipos de imágenes más allá de imágenes bmp. Se podría crear implementaciones para formatos .png y .jpeg. Otra mejora que podría aumentar el rendimiento del programa es permitir que el mensaje sea comprimido previamente a su ocultamiento, de esta manera se reduciría el tamaño del archivo y se permitiría poder esconder el mismo mensaje, en imágenes de tamaño más pequeño. Una extensión que sería positiva por temas de seguridad sería incluir un mecanismo de detección de modificación. Se podría implementar algún tipo de hash antes de realizar el esteganografiado y verificarlo después de la extracción. Esto permitiría saber si la imagen esteganografiada fue modificada después del ocultamiento del mensaje. Por último, una extensión que nos podría haber sido útil durante el desarrollo del trabajo práctico hubiera sido una detección de esteganografiado inverso. La idea de esta extensión es que dado un archivo podría realizar un análisis de las imágenes y detectar si una imagen tiene un mensaje oculto.