Criptografía y Seguridad *(72.04)*

TRABAJO PRÁCTICO DE IMPLEMENTACIÓN:

ESTEGANOGRAFÍA

1. Objetivos

- Introducirlos en el campo de la esteganografía y sus aplicaciones.
- Experimentar con métodos de ocultamiento de información en archivos de tipo "bmp", analizando ventajas y desventajas de cada uno.
- Implementar y analizar un algoritmo descripto en un documento científico.

2. Introducción

La ***esteganografía*** (del griego στεγανοζ *steganos, encubierto u oculto* y γραπηοζ *graphos, escritura*) es la ciencia que se ocupa de la manera de **ocultar** un mensaje.

La existencia de un mensaje u objeto es ocultada dentro de otro, llamado **portador o camuflaje**. El objetivo es proteger información sensible, pero a diferencia de la criptografía que hace ininteligible dicha información, la esteganografía logra que la información pase completamente desapercibida al ocultar su existencia misma.

La criptografía y la esteganografía se complementan. Un mensaje cifrado mediante algoritmos criptográficos puede ser advertido por un intruso. Un mensaje cifrado que, además, ha sido ocultado mediante algún método de esteganografía, tiene un nivel de seguridad mucho mayor ya que los intrusos no pueden detectar su existencia. Y si por algún motivo un intruso detectara la existencia del mensaje, encontraría la información cifrada.

La esteganografía tiene un origen muy antiguo. Ya Heródoto en el año 440 aC narra la historia de un mensaje escrito en una tablilla que es cubierto con cera para pasar desapercibido ante el enemigo. El mensaje puede así llegar a su destino, siendo develado por sus receptores al quitar la cera.

En la era digital, el interés se renueva por sus múltiples aplicaciones.

3. Consigna

Realizar un programa **stegobmp** en lenguaje C o Java que efectúe las siguientes operaciones:

- Oculte un archivo cualquiera en un archivo de extensión ".bmp", mediante un método de esteganografiado elegido, con o sin password.
- Descubra un archivo oculto en un archivo de extensión ".bmp" que haya sido previamente esteganografiado con uno de los métodos provistos.
- Estegoanalice un archivo de extensión ".bmp" para determinar si tiene un archivo incrustado, con qué algoritmo, y lo extraiga correctamente.

Sólo se permite hacer el programa en los lenguajes C o Java.
En el caso de usar lenguaje C, usando la librería openssl
En el caso de usar java, usando java 21.03 y javac 21.03.
Todo se probará por línea de comandos en WSL2 con Ubuntu 22.04.3 LTS

4. Archivos ".BMP"

El formato BMP es un formato de archivos de imagen bastante simple. Consta de dos partes:

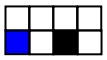
1. Encabezado → de 54 bytes.
2. Cuerpo → de tamaño variable.

El encabezado contiene información acerca del archivo: tamaño de archivo, ancho de imagen, alto de imagen, bits por pixel, si está comprimido, etc.

IMPORTANTE: Considerar la versión V3 de archivos BMP. Es la más común. No hace falta considerar otras versiones que puedan tener otro tamaño y datos de encabezados.

En el cuerpo del archivo bmp, están los bits que definen la imagen propiamente dicha. La imagen se lee de abajo hacia arriba y de izquierda a derecha. Si la imagen es de 24 bits por píxel, la distribución es: 8 primeros bits para azul, 8 bits para verde, y 8 bits para rojo.

IMPORTANTE: Observar que si bien los canales se conocen como RGB (Red Green Blue), el orden real en que se leen los bytes del archivo es BGR (blue, green, red).



Ejemplo: La imagen de la izquierda representa un bmp de 4 pixeles por 2 pixeles.

El primer píxel leido es el azul, luego el blanco, luego el negro.

Si la imagen es de 24 bits por píxel, el píxel azul tiene la siguiente estructura:

1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

En este trabajo práctico sólo interesa usar el cuerpo del archivo bmp como portador, para ocultar allí la información a esteganografiar. Por este motivo, sólo se utilizarán archivos .bmp que tengan las siguientes características:

- Imagen de 24 bits por pixel. Esto es, por cada pixel, se tienen 3 bytes donde ocultar información.
- Imagen bmp sin compresión. Para ello controlar el parámetro de compresión del encabezado.

5. Detalles del programa stegobmp

5.1. Ocultamiento de un archivo en un .bmp.

El programa debe recibir como parámetros:

- **`-embed`**

  Indica que se va a ocultar información.

- **`-in`** file

  Archivo que se va a ocultar.

- **`-p`** bitmapfile

  Archivo bmp que será el portador.

- **`-out`** bitmapfile

  Archivo bmp de salida, es decir, el archivo bitmapfile con la información de file incrustada.

- **`-steg <LSB1 | LSB4 | LSBI>`**

  algoritmo de esteganografiado: LSB de 1bit, LSB de 4 bits, LSB Enhanced

Y los siguientes parámetros opcionales:

- **`-a <aes128 | aes192 | aes256 | 3des>`**
- **`-m <ecb | cfb | ofb | cbc>`**
- **`-pass password`** (password de encripcion)

Ejemplo 1:

Esteganografiar el archivo de texto “mensaje1.txt” en el archivo portador “imagen1.bmp” obteniendo un archivo “imagenmas1 .bmp” mediante el algoritmo LSB Improved, con encripción 3DES en modo CBC con password “oculto”

```
$stegobmp -embed -in "mensaje1.txt" -p "imagen1.bmp" -out "imagenmas1.bmp" -steg LSBI -a 3des -m cbc -pass "oculto"
```

Ejemplo 2:

Esteganografiar el archivo de imagen “mensaje1.txt” en el archivo portador “imagen1 .bmp” obteniendo un archivo “imagenmas1.bmp” mediante el algoritmo LSB Improved, **sin encripción**

```
$stegobmp -embed -in "mensaje1.txt" -p "imagen1.bmp" -out "imagenmas1.bmp" -steg LSBI
```

IMPORTANTE: No se puede encriptar/desencriptar sin **password**. Si este dato no está, sólo se esteganografia.

Son válidas en cambio las siguientes opciones:

- indicar **algoritmo** y **password** pero no modo: Se asume CBC por default.
- Indicar **modo** y **password** pero no algoritmo: Se asume aes128 por default.
- Indicar **sólo password:** Se asume algoritmo aes128 en modo CBC por default.

5.2.Extraer de un archivo .bmp un archivo oculto.

- **-extract**

  Indica que se va a extraer información.

- **-p bitmapfile**

  Archivo bmp portador

- **-out file**

  Archivo de salida obtenido

- **-steg <LSB1 | LSB4 | LSBI>**

  algoritmo de esteganografiado: LSB de 1bit, LSB de 4 bits, LSB Improved

Y los siguientes parámetros opcionales:

- **-a <aes128 | aes192 | aes256 | 3des>**
- **-m <ecb | cfb | ofb | cbc>**
- **-pass password (password de encripcion)**

Ejemplo:

Extraer el archivo de texto "mensaje1.txt" del archivo portador "imagenmas1.bmp" ocultado mediante el algoritmo LSB Improved, con encripción 3DES en modo CBC con password "oculto"

```
$stegobmp -extract -p "imagenmas1.bmp" -out "mensaje1" -steg LSBI -a 3des -m cbc -pass "oculto"
```

5.3.Algoritmos de Esteganografiado.

Ocultamiento sin encripción:

Antes de ocultar el archivo propiamente dicho, con cualquiera de los algoritmos, ocultar su tamaño.

Después de ocultar el tamaño y el archivo propiamente dicho, con cualquiera de los algoritmos, ocultar su extensión (".png", ".jpg", ".txt", ".html", etc)

La extensión debe comenzar con '.' Y terminar con '\0'.

Es decir, se esteganografía:

Tamaño real || datos archivo || extensión

Y el total de datos a esteganografiar es:

4 (del tamaño) + longitud archivo + e (extensión)

Siempre se sabe que los primeros 4 bytes (DWORD size) corresponden al tamaño de archivo.

Siempre se sabe que después de los n bytes del archivo viene un punto y los n ascii de la extensión, terminando en '\0'.

Ocultamiento con encripción:

Si se eligió encriptar antes, se procede de la siguiente manera:

- Se obtiene la secuencia de bytes correspondiente a **Tamaño real || datos archivo || extensión.**
- Dicha secuencia se encripta con el algoritmo , modo y password.
- Se esteganografia el tamaño del cifrado y a continuación la secuencia cifrada.

Es decir, se esteganografía:

Tamaño cifrado || encripcion(tamaño real || datos archivo || extensión**)**

Y el total de datos a esteganografiar es:

4 (del tamaño del cifrado) + tamaño del cifrado

Para extraer, siempre se sabe que los primeros 4 bytes (DWORD size) corresponden al tamaño de cifrado. Con dicho tamaño, y algoritmo modo y password que se envian por argumento al programa se descifra.

Una vez hecha la desencripción se obtiene el tamaño real del archivo, se lee esa cantidad de bytes y se toma la extensión (hasta el '\0').

Ejemplo:

Si el tamaño real de "saco.txt" es 414 bytes, y se eligio encripcion en 3Des Cbc.

Entonces se encripta:

414 || datos archivo || ".txt\0"

Que da un total de 4 + 414 + 5 = 423 bytes.

No es múltiplo de bloque así que requiere padding. Por lo tanto se encripta 423 + padding.

Luego, se esteganografia:

(423+padding) || encripcion(414||datos archivo||".txt\0")

Es decir, se ocultan **4 (del tamaño del cifrado) + (423 + padding)** bytes.

1. LSB1

Inserción en el bit menos significativo (Least Significant Bit Insertion - LSB)

Es el método más simple y consiste en insertar información sustituyendo el bit menos significativo de cada byte del archivo portador por un bit del mensaje.

En el caso de archivos de imagen ".bmp" consideraremos que cada bit del mensaje se inserta en el bit menos significativo de la **componente** (cada pixel tiene un componente rojo, un componente verde y un componente azul, cada uno de 8bits).

Como se observa, el primer bit del mensaje a ocultar se inserta en el bit menos significativo de la primer componente del primer pixel, el segundo bit del mensaje a ocultar se inserta en el bit menos significativo de la segunda componente del primer pixel, el siguiente, en la tercer componente del primer pixel. Entonces para guardar un byte, se requieren 2 pixeles completos y 2 componentes del siguiente pixel.

Importante: Si el archivo bmp no puede albergar el archivo a ocultar completo, deberá mostrarse un mensaje de error, en el que se indique cuál es la capacidad máxima del archivo bmp portador.

2. LSB4

Inserción en los cuatro bits menos significativos

Es una variante del anterior, donde 4 bits del mensaje se ocultan en los 4 bits menos significativos de la componente del pixel que corresponda.

Importante: Si el archivo bmp no puede albergar el archivo a ocultar completo, deberá mostrarse un mensaje de error, en el que se indique cuál es la capacidad máxima del archivo bmp portador.

3. LSB Improved

Es una variante de los anteriores, propuesta por Majeed y Sulaiman en el documento **"An Improved LSB Image Steganography Technique using bit-inverse in 24 bit colour image"**.

Dicho documento se encuentra disponible para descargar en:

https://www.jatit.org/volumes/Vol80No2/16Vol80No2.pdf

Los bytes del bmp deberán usarse en el orden en que se lee físicamente el archivo. Es decir, después del encabezado, el primer byte es azul, el siguiente es verde, el tercero rojo. Se usarán entonces en ese orden.

Para guardar el registro de los cambios realizados, se guardará un 1 o un 0 indicando qué patrón de cambio se utilizó. Estos bits se guardarán usando LSB1 **antes** de los bits correspondientes al secreto. Es decir que, al total de datos a esteganografiar se le sumarán 4 bits (necesitando así 4 bytes más para ocultarlos).

Así, si por ejemplo se decidiera que los bits a invertir corresponden a los patrones '00' y '10', se guardará 1010 **en los primeros 4 bytes de la imagen** consecutivos con **LSB1 normal** (no se invierten, y se usan los 3 canales R, G y B.)

(1 indica que "00" tuvo cambio, 0 indica que "01" no tuvo cambios, luego 1 porque "10" sí tuvo cambios, y finalmente 0 porque "11" no tuvo cambios)

Importante: Si el archivo bmp no puede albergar el archivo a ocultar completo, deberá mostrarse un mensaje de error, en el que se indique cuál es la capacidad máxima del archivo bmp portador.

5.4. Otras consideraciones.

- **Respecto de cómo guardar el tamaño del bloque.**

Suponiendo que el tamaño del bloque a ocultar es 17,118.

Ese número ocupa 4 bytes.

Puede guardarse en Little Endian:

DE	42	00	00

O puede guardarse en Big Endian:

00	00	42	DE

Los archivos que les entregaremos tendrán el tamaño guardado en forma Big Endian.

- Si el archivo a ocultar requiriera padding usar el padding por defecto de openssl que es PKCS5.
- El archivo que se encripta es el que se quiere ocultar. Se encripta completo. Es decir, aún si fuera un bmp, se encripta desde el byte 0 hasta el último, cabecera incluida.
- Para la generación de clave a partir de una password, deberán usar el algoritmo PBKDF2, usando SALT fijo 0x0000000000000000 (8 bytes en 0, expresados en hexa)
- Los bmps utilizados deben ser de **24 bits** (8bits para rojo, 8bits para verde y 8bits para azul).
- Los bmps utilizados **no deben** tener compresión.
- En algunos archivos bmp, se completa con "padding" para lograr el tamaño de múltiplo de 4 bytes en las filas de píxeles. Para el esteganografiado usen todo el bloque completo de los pixeles + bytes de padding.
- Para modos de feedback considerar una cantidad de 8 bits si es CFB y 128 si es OFB.
- El algoritmo 3DES corresponde a Triple Des, con tres claves distintas (`des_ede3` en openssl).

Nota: El algoritmo PBKDF2 retorna siempre una cadena de la cual hay que separar key e iv.

Equivalencias OpenSSL / JCE para modos de Feedback:

	OpenSSL	JCE
CFB	`EVP_aes_128_cfb()`,	"AES/CFB/NoPadding" (corresponde a 128 bits de feedback)
	`EVP_aes_128_cfb1()`	NO tiene
	`EVP_aes_128_cfb8()`	"AES/CFB8/NoPadding" (corresponde a 8 bits de feedback)
OFB	`EVP_aes_128_ofb()`	"AES/OFB/NoPadding" (corresponde a 128 bits de feedback)
	NO tiene para 1 bit	NO tiene para 1 bit
	NO tiene	"AES/OFB8/NoPadding" (corresponde a 8 bits de feedback)

6. Estegoanálisis

La cátedra proveerá de 4 archivos con información oculta. De ellos, uno de los archivos contendrá un archivo ocultado mediante LSB1, otro mediante LSB4 y otro mediante LSBI. Un cuarto archivo ocultará información de otra forma que habrá que descubrir.

En los que están esteganografiados con los métodos LSB1, LSB4, LSBI, se sigue el formato especificado previamente:

tamaño + archivo + extensión

Habrá que obtener los archivos ocultos (estegoanálisis) descubriendo de qué manera fueron ocultados.

En general, el análisis que se puede hacer es:

- por conocimiento de archivo portador
- por repetición de archivo portador
- estadístico
- de tamaño de los archivos
- etc.

IMPORTANTE:

Se recomienda, para este análisis, usar algún editor de archivos binarios (hex editor neo o similar) que permita hacer comparaciones de los archivos.

7. Cuestiones a analizar

I. Discutir los siguientes aspectos relativos al documento.

a) Organización formal del documento.

b) La descripción del algoritmo.

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

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.

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.

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

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?

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

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

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?

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

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

8. Organización de los grupos

El trabajo será realizado en grupos de máximo 4 integrantes. Se inscribirán los grupos a través del campus.

9. Entrega

La fecha de entrega es el día **10 de noviembre**.

Cada grupo entregará el código en C o Java, junto con los archivos para obtener el ejecutable y la documentación correspondiente al uso del programa a través de una actividad específica en el campus.

Además presentarán un informe con la solución correspondiente al estegoanálisis de los archivos que se le entregarán oportunamente al grupo y con el análisis de cada una de las cuestiones planteadas en el punto 7.

IMPORTANTE:

El README debe indicar claramente todos los pasos para lograr el ejecutable, y deben estar todos los archivos necesarios para hacerlo: makefile, pom.xml, etc.

Si al intentar obtener el ejecutable encontramos problemas, lo deberán solucionar en forma inmediata. Aún así, se les descontarán puntos.

10. Material de consulta recomendado

Hay mucho material en internet sobre el tema.

Es obligatorio leer:

Sobre el método **que para este TP denominamos LSBI**:

El documento **"An Improved LSB Image Steganography Technique using bit-inverse in 24 bit colour image"** cuyos autores son Mohammed Abdul Majeed y Rossilawati Sulaiman de la Universidad Kebangsaan Malaysia, de Malasia. Dicho documento se encuentra disponible para descargar en:

https://www.jatit.org/volumes/Vol80No2/16Vol80No2.pdf

Sobre **archivos bmp**:

https://docs.microsoft.com/en-us/windows/win32/api/wingdi/ns-wingdi-bitmapfileheader?redirectedfrom=MSDN

Otros documentos que son útiles para la comprensión del tema o para contestar las preguntas:

- Cummings, Jonathan y otros: **Steganography and Digital Watermarking**. Disponible en: https://slideplayer.com/slide/4615831/
- Gómez Cárdenas, Roberto: **Esteganografía**. Disponible en: http://www.cryptomex.org/SlidesCripto/Estegano.pdf
- Johnson, Neil F. y Jajodia, Sushil : **Exploring Steganography. Seeing the Unseen**. Disponible en: http://www.fim.uni-linz.ac.at/lva/Rechtliche_Aspekte/2001SS/Stegano/leseecke/steganography%20seeing%20the%20unseen%20by%20neil%20f.%20johnson.pdf