Trabajo Práctico Especial Nº2
Criptografía y Seguridad
Esteganografía

ITBA - Grupo 4

Autores:
Laurent Georget
José Ignacio S. Galindo

Introducción

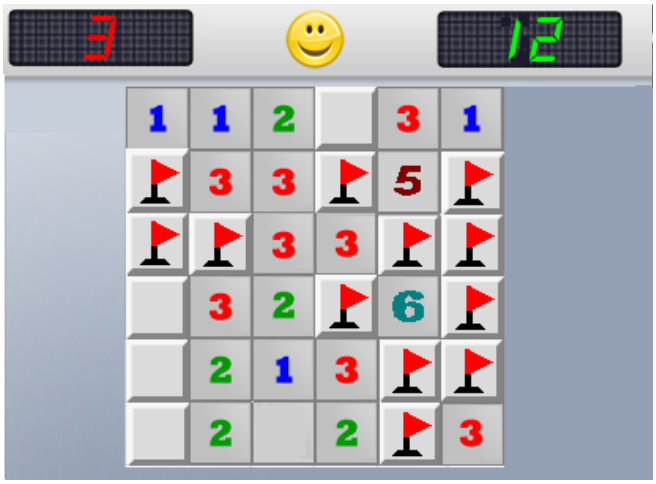
Este trabajo consistió en implementar un mecanismo de estenografía digital usando como portador imágenes de formato BMP. La ventaja de BMP es que es un formato no comprimido, eso permite hacer manipulación a nivel de bits sin corromper el formato de la imagen.

El programa que resultó del proyecto, **stegobmp**, permite esconder y extraer un archivo de una imagen BMP. Usa distintos algoritmos de esteganografía y permite además el cifrado del archivo que esconder antes de la operación de esteganografía propiamente dicho. El programa fue realizado en lenguaje C, y utiliza la biblioteca *openssl* para las operaciones de cifrado y descifrado.

Estegoanálisis

Fueron provistos 4 imágenes bmp con información oculta. En función de esos archivos y sin conocer de qué forma había sido ocultada la información, mediante un mecanismo de prueba y error se obtuvieron los siguientes resultados:

- El archivo *hugo5.bmp* contenía la siguiente imagen con formato PNG. Esta imagen fue extraída utilizando el modo LSB4 .



- El archivo *medianocheenparis1.bmp* contenía un documento PDF con el siguiente contenido.

“al .png cambiarle la extension por .zip y descomprimir “

El documento fue extraído utilizando el modo LSBE.

- Una vez cambiada la extensión de archivo extraido de la imagen hugo5.bmp y de descomprimirlo, se obtuvo un arhivo sol1.txt con el siguiente contenido:

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

- El texto hace referencia a la imagen PNG extraída de *hugo5.bmp*. Reemplazando cada cuadrito conteniendo una mina por un “1” y los demás por un “0” (conociendo las reglas del juego), se obtuvo:

00 0100
10 0101
11 0011
00 0101
10 0011
10 0010

Y agregando los bits “01” al principio, se obtuvieron seis bytes, que son los códigos ASCII de los caracteres “Des Ecb”. Así se encontró el algoritmo y el modo de cifrado.

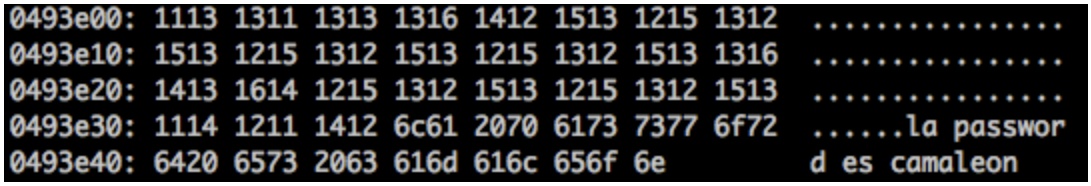
- Como la password no se encontraba en ningun lado, se consideró la idea de analizar los archivos de la carpeta con un editor hexadecimal de la siguiente manera:

xxd <file> | grep “password”

- Al fin, la cadena “password” apareció en uno de los archivos: la imagen *lifeofpi.bmp* que no parecía tener archivo oculto (no se podía levantar un contenido, con ninguno de lo tres algoritmos). Al ejecutar el comando siguiente,

xxd archivo | tail -n 5

apareció como resultado lo que muestra la imagen a continuación:



- Sabiendo que la contraseña era *camaleon*, se extrajo de la imagen eclipse.bmp un video con extensión WMV, que resultó ser el secreto oculto en las imágenes.

La serie concreta de comandos que se ejecutaron para descubrir el secreto fueron los siguientes:

```
$> cd src
$> ../../src/stegobmp --extract -p hugo5.bmp --out out_hugo5 --steg LSB4
$> ../../src/stegobmp --extract -p medianocheenparis1.bmp --out out_medianocheenparis1
--steg LSBE
$> cp out_hugo5.png out_hugo5.zip
$> unzip out_hugo5.zip
$> cat sol1.txt
$> for i in *; do; echo $i; xxd $i | grep pass -A4 -B4; done
$> xxd lifeofpi.bmp | tail -n 5
$> ../../src/stegobmp --extract -p eclipse.bmp --out out --steg LSB1 --pass camaleon -m
ecb -a des
```

Cuestiones Analizadas

Una vez realizado el programa, se resolvieron las siguientes 9 cuestiones:

1. **Para la implementación del programa stegobmp se pide que la ocultación comience en el primer componente del primer pixel. ¿Sería mejor empezar en otra ubicación? ¿Por qué?**
   Sería mejor empezar en otra ubicación, porque si el *offset* es siempre el mismo, un atacante puede rápidamente darse cuenta si hay un archivo oculto o no.
   El problema de empezar en otra ubicación es que hay que encontrar una forma de compartir esta ubicación entre el que envía el mensaje y el que lo recibe. Además, empezar en otra ubicación desperdicia espacio, a menos que se considere el arreglo de pixeles de la imagen como un buffer circular.

2. **¿Qué ventajas podría tener ocultar siempre en una misma componente? Por ejemplo, siempre en el bit menos significativo de la componente azul.**
   Ocultar siempre en una misma componente no traería ninguna ventaja con respecto a una mejor ocultación. Ya que en primer lugar si se usara una sola componente se

estaría perdiendo espacio, y en segundo lugar, el ocultamiento de la información podría ser más notorio en la imagen portadora al haber modificado sólo la componente azul, por ejemplo.

3. **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.**
   El *tradeoff* es "calidad de la ocultación" (que tan difícil es distinguir una imagen portadora de otra que no lo es) contra "cantidad de bytes del portador necesarios para ocultar un byte".
   Se estenografió una misma imagen BMP con los tres modos implementados y se obtuvo los resultados detallados en el siguiente cuadro comparativo:

Comparación	Tamaño necesario del portador	Calidad de la ocultación	Proporción de bits modificados (promedio)
LSB1	grande (8 veces el tamaño del archivo que ocultar)	indistinguible para un ojo humano, pero se notan diferencias con imagenes comunes (histogramas, distribución de colores entre pixeles adyacentes...)	tamaño del portador / 16
LSB4	pequeño (2 veces el tamaño del archivo que ocultar)	indistinguible para un ojo humano no experto, pero la análisis automática muestra diferencias importantes con una imagen normal	tamaño del portador / 4
LSBE	muy grande (suficientemente grande para contener 8 veces más de bytes >= 254 que el tamaño del archivo que ocultar)	indistinguible tanto para un humano como para una análisis automática, salvo con una comparación a nivel de bits con la imagen de origen	tamaño del portador * proporción de bytes >= 254 / 16

   Como se puede apreciar, el algoritmo LSB4 es mejor teniendo en cuenta que necesita menos espacio en la imagen portadora para ocultar un mensaje.
   Por el otro lado y teniendo en cuenta el otro criterio se puede ver como el algoritmo LSBE es superior con respecto a la calidad de ocultación y porción promedio de bits modificados.
   Se debe aclarar que considera que en promedio se modifican la mitad de los bits que se quieren ocultar, ya que esos son los que efectivamente se cambian.

4. **Para la implementación del programa stegobmp se pide que la extensión del archivo se oculte después del contenido completo del archivo. ¿por qué no conviene ponerla al comienzo, después del tamaño de archivo?**
   Si la extensión del archivo se escondiera al comienzo, ésta sería demasiado visible, al estar siempre en la misma ubicación. Alguien interesado en conocer lo que en el portador se oculta, podría rápidamente descubrir sin mucho esfuerzo que tipo de archivo es el que se esconde, lo que posiblemente le facilitaría su recuperación.

5. **Explicar detalladamente el procedimiento realizado para descubrir qué se había ocultado en cada archivo y de qué modo.**
   Explicado detalladamente en la sección de Estegoanálisis.

6. **¿Qué se encontró en cada archivo?**
   hugo5.bmp: una imagen out.png. Cambiando la extensión de esta imagen y descomprimiéndola, se encuentra un sol1.txt que explica que hay un archivo *.wmv* encriptado en uno de los archivos y que da la clave del código secreto de out.png
   Con ese código, se puede conocer el algoritmo y modo de encriptación del *.wmv*.
   medianocheenparis1.bmp: tiene oculto un documento PDF que explica que hay que cambiar la extensión de la imagen PNG obtenida de *hugo5.bmp* de *.png* a *.zip*.
   lifeofpi.bmp: No hay nada esteganográfico con uno de los algoritmos que usamos. No se puede obtener un tamaño correcto. Pero en los últimos bytes del archivo está la cadena "la password es camaleón"
   eclipse.bmp: hay un archivo *.wmv* encriptado que se puede recuperar conociendo la password, el algoritmo y el modo de encriptamiento.

7. **Algunos mensajes ocultos tenían, a su vez, otros mensajes ocultos. Indica cuál era ese mensaje y cómo se había ocultado.**
   La imagen PNG que estaba oculta en *hugo5.png* contenía después de los bytes de la imagen un archivo zip entero que solamente se podía descubrir intentando **unzipear** (o dándose cuenta que el PNG era más grande que debía ser).

8. **Uno de los archivos ocultos era una porción de un video, donde se ve ejemplificado una manera de ocultar información ¿cuál fue el portador?**
   El portador de la porción del video donde se ve ejemplificado una manera de ocultar información era la imagen *eclipse.bmp*. Dicho video, además de estar oculto con el modo LSB1 también estaba encriptado utilizando DES con ECB.

9. **¿De qué se trató el método de estenografiado que no era LSB? ¿Es un método eficaz? ¿Por qué?**
   Se detectaron dos métodos de estenografiado que no eran LSB:
   a. **concatenar un archivo .ZIP después de un archivo .PNG:** No es muy eficaz porque es fácil para un atacante darse cuenta que el tamaño del PNG no corresponde con su contenido, luego extraer los bytes ubicados a continuación del contenido de la imagen y adivinar el tipo del archivo.
   b. **escribir la información en texto plano dentro de un archivo binario:** No es muy eficaz ya que con un editor hexadecimal se puede obtener muy fácilmente la cadena oculta. Incluso se puede hacer de manera automatizada, buscando con una expresión regular una cadena ASCII por ejemplo, la información.
10. **¿Qué mejoras o futuras extensiones harías al programa stegobmp?**
    Posibles mejoras podrían ser la siguientes:
    - soportar otras versiones de imágenes BMP como portadores (en este trabajo sólo se trabajó con la versión 3).
    - soportar otros tipos de imágenes como portadores, de manera que poder usar

no sólo imágenes BMP para guardar datos ocultos.
    - soportar otros tipos de archivos como portadores, como por ehemplo arhivos de audio o de video.
    - soportar otros tipos de esteganografía fuera de los LSB implementados
    - soportar otros tipos de encriptado. Por ejemplo se podría implementar cifrado asimétrico.
    - soportar detección y extracción **automática** de un archivo portador

Fuentes

- Efficient method of audio steganography by modified LSB algorithm and strong encryption key with enhanced security" http://www.jatit.org/volumes/research-papers/Vol5No6/15Vol5No6.pdf
- Sobre archivos bmp: (http://msdn.microsoft.com/en-us/library/windows/desktop/dd183374(v=vs.85).aspx)
- Cummings, Jonathan y otros: Steganography and Digital Watermarking. (http://www.cs.bham.ac.uk/~mdr/teaching/modules03/security/students/SS5/Steganography.pdf)
- Isaza, Gustavo A. y otros: Análisis de técnicas esteganográficas y estegoanálisis en canales encubiertos, imágenes y archivos de sonido. En (http://vector.ucaldas.edu.co/downloads/Vector1_3.pdf)
- Gómez Cárdenas, Roberto: Esteganografía. Disponible en: (http://www.cryptomex.org/SlidesCripto/Estegano.pdf)
- Johnson, Neil F. y Jajodia, Sushil : Exploring Steganography. Seeing the Unseen. (http://www.creangel.com/papers/steganografia.pdf)
- Gupta, Shilpa; Gujral, Geeta y Aggarwal, Neha: Enhanced Least Significant Bit algorithm For Image Steganography. (www.ijcem.org/papers072012/ijcem_072012_08.pdf)