Criptografía y Seguridad

Esteganografía

Grupo 1

- Kulesz, Sebastián
- López, Noelia Belén
- Moreno, Juan Pablo
- Vera, J. Sebastián

Introducción

Con la realización de este trabajo se busca poder desarrollar un programa capaz de ocultar (y descubrir) información en archivos de imágen (.bpm), siguiendo el concepto de esteganografía.

Fueron utilizados los métodos LSB1, LSB4 y LSBE propios de la esteganografía y pudieron ser, además, encriptados (y desencriptados).

En el siguiente informe se encuentran resueltas las preguntas establecidas por la cátedra dentro de las cuales se realiza un análisis esteganográfico con los archivos provistos.

Resolución

Para la implementación del programa stegobmp se pide que la ocultación comience en el primer componente del primer píxel. ¿Sería mejor empezar en otra ubicación? ¿Por qué?

Previo a la matriz de píxeles en la que se encuentran los datos de las imágenes, en un archivo .bpm, están los headers. Estos no pueden ser alterados ya que contienen información detallada sobre la imágen de cómo *mostrarla en pantalla.* Es posible que el cambio en uno sólo de los bits altere las condiciones.

Empezar desde el píxel K (en la matriz de pixeles) tampoco es conveniente ya que, si bien no produciría una alteración en las propiedades de la imagen, reduciría el espacio disponible para ocultar el archivo.

Es preferible empezar en el primer píxel y comenzar *escondiendo* el próximo byte y, así, generar un proceso secuencial.

¿Qué ventajas podría tener ocultar siempre en una misma componente? Por ejemplo, siempre en el bit menos significativo de la componente azul.

Cada vez que modificamos un bit de imagen, en este caso para ocultar información, modificamos la imágen en sí y por ende como se muestra en pantalla. Ocultar siempre en una misma componente resultará en que la imagen varíe menos (menos manchada), en el color de cada píxel, con respecto a la original haciendo menos notable a quién conozca el archivo original, que está siendo utilizado para ocultar información. A lo sumo será toda la imagen *más azulada*.

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

	Ventajas	**Desventajas**
LSB1	- *Mayor fidelidad* con la imagen original	- Mayor requerimiento de espacio (comparado con LSB4) - Se nota la variación en cualquier secuencia de bytes
LSB4	- Mayor espacio disponible para ocultar un archivo	- Imagen resultante más *manchada*
LSBE	- Imágen mucho menos manchada y mucho más fiel	- Depende de que el archivo portador tenga los bytes 254 y 255 (archivos pequeños pueden

		no tenerlos)

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?

Teniendo en cuenta que se utilizan extensiones conocidas, al saber primero la extensión, se puede tratar de obtener información sobre los datos leídos, resultando más fácil descubrir si existe algún mensaje *esteganografiado.*

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

Lo primero que se hizo fue crear un pequeño script que corriera la extracción de archivos con los 3 algoritmos propuestos. Se asumió, en primer lugar que, de tener información oculta, las imágenes no serían contenedores de archivos encriptados.

Las imágenes *avatar.bmp*, *quito.bmp* y *budapest.bmp* revelaron tener información oculta al ser descubiertos con los algoritmos *LSB1, LSBE* y *LSB4* respectivamente. Se nombró a cada archivo descubierto *avatar-descubierto, quito-descubierto, budapest-descubierto.*

Siguiendo la recomendación de la cátedra, se utilizó *hexdump* para analizar el contenido del archivo *buenosaires.bmp.*

¿Qué se encontró en cada archivo?

El archivo *budapest-descubierto* fue guardado automáticamente como archivo *.png,* el mismo contenía una imagen de un cuadro del famoso juego *buscaminas*. El archivo *quito-descubierto* fue automáticamente guardado como archivo *.pdf*, el mismo contenía el texto *"la password es sorpresa"*. El archivo *budapest-descubierto* fue automáticamente guardado como archivo *.wmv,* el mismo no pudo ser reproducido; se asumió que el contenido estaba encriptado.

Así mismo, el final del archivo *buenosaires.bmp* contenía una serie de caracteres que se podían leer como *"al .png cambiar extensión por .zip y descomprimir"*. Se procedió a cambiar la extensión de la imagen del *buscaminas* para poder descomprimirlo. Al hacerlo se encontró dentro un archivo de texto con la siguiente instrucción:

"cada mina es un 1.
cada fila forma una letra.
Los ascii de las letras empiezan todos en 01.
Asi 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."

Con esa información y sabiendo que el archivo *budapest-descubierto* contenía un archivo *.wmv* ya se podría obtener la porción del video.

Si bien también se creó un *script* que corriera todos los métodos de desencripción (*aes128, aes192, aes256, des*) con todos sus modos (*ecb, cfb, ofb, cbc*), se procedió a seguir las instrucciones para el entender el mensaje oculto dentro del buscaminas. El mismo confirmaba que el método de encripción había sido *AES* con modo *ECB*.

De esta manera se pudo revelar el archivo *.wmv* de manera que fuera posible de ser ejecutado. El mismo era una porción de una serie (doblada al español) sobre mensajes encriptados en tejidos.

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

La imagen *avatar.bmp* contiene una imagen *png* oculta que a su vez podía ser entendido como un archivo comprimido con un archivo *.txt* dentro. Así mismo, se puede entender que la imagen del buscaminas, en sí, tiene, también, un mensaje oculto.

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 fue la imagen *budapest.bmp.*

¿De qué se trató el método de estenografiado que no era LSB? ¿Es un método eficaz? ¿Por qué?

El método consta de agregar información al final del archivo. Si bien a nivel programático es bastante más sencillo, no es tan eficaz si se desea ocultar un archivo de manera inadvertida ya que sería muy sencillo poder recuperar esa información (con un lector hexadecimal).

A su vez, si la información oculta fuera demasiado grande, se vería una mancha al fondo de la imagen que podría rápidamente despertar sospechas.

¿Qué mejoras o futuras extensiones harías al programa stegobmp?

Los algoritmos planteados trabajan con los bits menos significativos del byte, por lo que se podría pensar en extender el programa para trabajar con archivos de audio (.wav) utilizando muestras del archivo en lugar de píxeles. Sería casi imperceptible un pequeño ruido en cada muestra, incluso podría *culparse* al grabador, o al reproductor, o incluso al ruido ambiente.

Se podría implementar LSB2 o LSB3 ya que tienen mayor capacidad que LSB1 y menor riesgo de *manchas* que LSB4.

Por otro lado, podrían utilizarse alteraciones a la hora de ocultar información como por ejemplo empezar desde el final hacia el comienzo, simplemente porque es menos intuitivo y eso dificultaría que fuera evidente que se esconde información; o se podŕian utilizar los bits en las posiciones primas o (im)pares.

Indicar ventajas y desventajas de haber elegido utilizar el padding de bytes de los archivos BMP, qué otra opción podría haberse elegido, y por cada una, ventajas y desventajas (desde el punto de vista de implementación y desde el punto de vista del esteganografiado).

La principal desventaja de utilizar el padding es que el mismo es de valor fijo que, al encontrarse modificado, deja al descubierto que la imagen es un archivo portador. Se podría pensar, sin embargo, controlar los headers para sólo permitir imágenes que no tengan padding y así evitar lo anterior, aunque eso reduciría el espacio de posibles portadores.

Conclusión

Se concluye, por consiguiente, que es relativamente muy sencillo ocultar información dentro de archivos de uso común como una imágen de manera casi imperceptible.

Por otro lado, queda en evidencia que, si bien el tamaño de la información a ser ocultada se ve restringido por el tamaño del portador y del método a utilizar, es posible utilizar varios portadores para transmitir información (un archivo encriptado en un portador y la clave para desencriptarlo en otro portador).

El tamaño es un factor limitante al igual que las *manchas* (o variaciones en la tonalidad original) son un factor que pone en evidencia el carácter de portador de un archivo; ambos factores son mutuamente excluyentes por lo que es necesario encontrar el equilibrio entre ambos.