Criptografía y Seguridad
Esteganografía

Giorgi Pablo (49222)
Perez De Rosso, Santiago (48274)
Zemín Luciano (48394)

2 de junio de 2011

Índice

1. Introducción **1**

2. Cuestiones a analizar **1**
2.1. Ejercicio 6.1 1
2.2. Ejercicio 6.2 2
2.3. Ejercicio 6.3 3
2.4. Ejercicio 6.4 3
2.5. Ejercicio 6.5 4
2.6. Ejercicio 6.6 5
2.7. Ejercicio 6.7 5
2.8. Ejercicio 6.8 5
2.9. Ejercicio 6.9 5

1. Introducción

En el presente documento se analizan cuestiones referentes al campo de la esteganografía.
Se elaboran conclusiones respecto al método en sí mismo, y cómo se lo podría mejorar. Además se lo combina con conceptos de criptografía para complementarlo y aumentar su seguridad.

2. Cuestiones a analizar

2.1. Ejercicio 6.1

Respecto de dónde comezar a guardar el archivo a ocultar se debe tener en cuenta un punto importante.
En la encriptación, comunmente se conoce que la información existe, pero se desea ocultarla. En la esteganogrfía, por el contrario, lo que se desea ocultar es la existencia misma de la información. Por tal motivo, a menos que se use esteganografía y encriptación en conjunto, si se sospechara de la existencia de la información no sería difícil que eventualmente se pudiera acceder a ella.
Por lo tanto, en un escenario de esteganogrfía puro (sin encriptar la información antes de ser ocultada), es altamente deseable que una vez que el archivo portador ya fue modificado, dicha modificación sea lo más imperceptible posible.
Partiendo de esta base se puede pensar que la mejor forma de guardar información dentro de un archivo sea comenzar en aquel lugar donde el cambio sufrido por el archivo portador sea mínimo. Se puede afirmar que por cada bit que se desea almacenar, la probabilidad de producir un cambio en el archivo es de un 50 %, dado que si se desea guardar un 0, hay un 50 % de chances de que en la posición a modificar haya un 0 (no habría cambio) o haya un 1 (habría cambio). Al guardar

un 1, el caso es análogo.
Por lo tanto, una buena estrategia para decidir dónde conviene comenzar a modificar el archivo es haciendo un análisis previo del mismo analizando tal cuestión.
Para verlo con un ejemplo simple. Se tiene que guardar la cadena 0011 en un archivo de 6 muestras de 2 bytes cada una de la siguiente manera:
01100101 11001001
00110001 11000010
00101011 00110011
01011000 00111101
10111001 10100110
00111100 01100110
Al tener el archivo portador 6 muestras de 2 bytes, y al tener que ocultar 4 bits, sólo se podría comenzar en las muestras 1, 2 o 3.
Para facilitar el análisis nos quedaremos con los bits que son factibles de ser modificados (los menos significativos de cada muestra), teniendo así la cadena 101100. Las 3 diferentes opciones para comenzar nos darían como resultado las cadenas:
001100 para la primera.
100110 para la segunda.
100011 para la tercera.
El porcentaje de cambio sufrido en cada una es de:
1/6 para la primera.
2/6 para la segunda.
4/6 para la tercera.
Lo que da como resultado que en el caso particular del ejemplo, la posición más conveniente donde comenzar a guardar los datos es a partir de la primera muestra.
Es importante aclarar que si se considera la posibilidad de comenzar a guardar la información en una posición que puede cambiar, también ésta debe ser almacenada en el archivo, dado que de lo contrario no sería posible recuperar la información oculta. Una opción sería ahora sí guardarla a partir de la primera muestra, dado que el ruido que puede producir una modificación tan pequeña como para sólo guardar un número con una posición es mucho menor probabilísticamente al que produce la modificación de tantas muestras seguidas como lo requiere guardar la información que se desea ocultar.

2.2. Ejercicio 6.2

Luego de esteganografiar un mismo archivo dentro de un mismo portador, habiendo obtenido 3 nuevos archivos de audio con la información, se los compara arribando a las conclusiones que se exponen a continuación.
En primer lugar se intentó buscar algún tipo de ruido en los archivos obtenidos luego de esteganografiar, o algún tipo de diferencias respecto del original.
Como no se encontró lo buscado, se procedió a analizar las pistas con el *Audacity* intentando analizar la misma cuestión pero ahora de forma gráfica. Tampoco se obtuvieron resultados positivos.
Por último, se buscaron diferencias entre los archivos abriendo cada uno de ellos con un editor hexadecimal, en particular se utilizó *0xED*. Mediante este método de análisis se lograron ver diferencias entre ellos como era de esperarse.
Cabe aclarar que se pudo ver que en este caso en particular, donde el archivo que se intentó ocultar era chico en tamaño, los métodos de esteganografiado resultaron efectivos, al no develar la existencia de la información. De hecho, si no se contara con otros archivos con los que comparar, no se podría decir que ninguno de ellos tiene información oculta adentro.

En cuanto a comparaciones del método se puede decir lo siguiente de cada uno de ellos:

Método	Ventajas	Desventajas
LSB1	Al producir un cambio en el archivo, el mismo es pequeño dado que sólo modifica 1 bit por muestra. Además, este bit es el menos significativo de la muestra, por lo que el cambio es mínimo.	Al modificar sólo 1 bit por muestra, necesita una mayor cantidad de muestras para lograr ocultar el archivo completo. Ésto no sólo ocasiona que el archivo portador tenga que ser más grande, sino que también una vez producido el cambio modifica una mayor cantidad de muestras.
LSB4	Al guardar 4 bits por muestra, la cantidad que necesita es menor que en LSB1. Lo que ocasiona que el archivo portador pueda ser de menor tamaño.	Al producir un cambio en el archivo, el mismo es más notorio que en LSB1, dado que a pesar de que modifica el byte menos significativo de la muestra, modifica 4 bits del mismo, en lugar de sólo 1.
LSBE	Al modificar sólo los bytes con valor 254 o 255, las modificaciones se encuentran más distribuídas dentro del archivo portador, por lo que sería más complicado detectar cambios en el mismo. Por otro lado, al cambiar sólo un bit los cambios son mínimos.	Por esta misma razón es que el archivo portador debe ser de un tamaño mayor que en los casos de LSB1 y LSB4. Por otro lado, dado que el método no distingue entre el byte más y menos significativo, por lo que podría modificar el más significativo, agregando así una cantidad grande de ruido.

Cuadro 1: Tabla de comparaciones de los métodos de esteganografiado analizados estableciendo ventajas y desventajas

2.3. Ejercicio 6.3

La cuestión a analizar referida a dónde conviene guardar el tipo de archivo es un punto importante pensando en cúales serían las posibles formas de descubrir que hay información oculta dentro de un archivo.
Una posibilidad sería analizar el archivo de audio intentando encontrar un header, o algún patrón que cumpla tal funcionalidad. El lugar más común donde se lo buscaría es en el comienzo de los datos del mismo.
Dado que las posibles extensiones de archivos son limitadas, no sería difícil encontrarlas si se las buscara en el lugar adecuado. Es por tal motivo que es preferible guardar la extensión al final y no al principio. La diferencia es que el tamaño, al ser un número, no podría ser distinguido como parte de los datos ocultos, mientras que lo mismo ocurre con los datos, pero las posibles extensiones sí pueden ser distinguidas.

2.4. Ejercicio 6.4

Para descubrir qué se había ocultado dentro de cada archivo, siempre teniendo en cuenta que se conocía que cada archivo ocultaba algo, se procedió de la siguiente manera:

En primer lugar se analizaron los archivos proporcionados .wav, viendo que 2 de ellos eran la misma canción. Sin conocimiento del algoritmo de esteganografiado usado en ninguno los 4 archivos, se procedió a aprovechar la semejanza del 6c.wav y 6d.wav para su comparación, dado que un diff byte a byte probablemente permitiría descifrar los algoritmos de esteganografiado utilizados en los mismos.
Luego de realizar la comparación byte a byte, se notó que gran parte de los archivos en los que diferían, poseían el valor FF o FE, lo cual apuntaba a que uno de los archivos estaba esteganografiado con el algoritmo LSBE. Por otra parte, el resto de los bytes en que diferían, eran siempre el byte de la izquierda de cada muestra, que en archivos .wav es el byte menos significativo. Además diferían en tan solo 1 bit, el menos significativo del byte. Ésto apuntaba a

que el otro archivo estaba esteganografiado con el algoritmo LSB1.
Una vez descifrados los dos algoritmos utilizados por los archivos, se procedió a probar ambos en el archivo 6d.wav para su desesteganografiado, de lo que resultó que el algoritmo indicado era LSBE. Luego, el algoritmo utilizado por el archivo 6c.wav era LSB1, lo cual se corroboró al desesteganografiarlo con dicho algoritmo.
Se obtuvo la información del archivo 6d.wav, que resultó ser un documento .pdf con una password. Por otra parte, la información del archivo 6c.wav resultó ser una imagen .png con un tablero del juego *Buscaminas*.
En tercer lugar, dado que se sabía que uno de los archivos restantes escondía información sin utilizar un algoritmo LSB, y dado que la cátedra sugirió analizar el archivo en cuestión con un editor hexadecimal o audacity, se procedió a analizar ambos archivos con un editor hexadecimal. En el archivo 6a.wav no se logró observar ningún tipo de información fácilmente relevante, pero en el archivo 6b.wav se logró descubrir que al final del mismo se encontraba la frase:

```
al .png cambiar extension por .zip y descomprmir.
```

Con la ayuda descubierta en el archivo anterior, y ya teniendo la imagen .png se procedió a hacer lo indicado. Se cambió la extensión a .zip, se descomprimió y se obtuvo un archivo .tex con más indicaciones.
El mismo decía:

```
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 256 bits y el modo
La password esta en otro archivo
Con algoritmo, modo y password hay un .wmv encriptado y oculto.
```

Siguiendo la ayuda se armó con el tablero que se encontraba en la imagen .png una matriz de 0 y 1 según donde había o no minas.
Se obtuvo la siguiente matriz, que convertida a código ASCII decía lo siguiente:

0100 0001 = 0x41 = A
0110 0101 = 0x65 = e
0111 0011 = 0x73 = s
0100 0101 = 0x45 = E
0110 0011 = 0x63 = c
0110 0010 = 0x62 = b

Por último, utilizando el método LSB4, la password obtenida anteriormente y el método de encriptación con el modo recién descubiertos se procedió a obtener la información del archivo 6a.wav. Lo que se obtuvo fue un video .wmv.

2.5. Ejercicio 6.5

Lo que se encontró en cada archivo fue explicado en el inciso anterior, pero se mostrará en un cuadro a continuación:

Archivo portador	Archivo recuperado
6a.wav	video .wmv
6b.wav	frase guardada en Hexa
6c.wav	imagen .png
6d.wav	documento .pdf

Cuadro 2: Tabla que indica qué se logró recuperar de cada uno de los archivos provistos por la cátedra

2.6. Ejercicio 6.6

El archivo en cuestión era un archivo de imágen .png, al cual se le pudo cambiar la extensión a .zip, y mediante el comando unzip, descomprimir un archivo de texto que incluía información relevante para poder seguir adelante con el desarrollo del trabajo.
La forma de realizar ésto es la siguiente: se utiliza el comando 'cat' y el operador de redirección >, por ejemplo, 'cat imagen.png comprimido.zip > imagenconcomprimido.zip'
Lo que realiza ésto, es concatenar ambos archivos y escribir la totalidad en otro archivo de extension .png.
La razón de por qué se puede visualizar la imagen y a la vez obtener el contenido del comprimido a partir de un mismo archivo, se debe a a que si bien para que la imagen no se corrompa, el header y contenido de la misma deben estar al comienzo del archivo, no ocurre lo mismo con los comprimidos .zip, dado que la utilidad unzip utilizada para descomprimirlo, no requiere que el header y contenido del mismo se encuentren al principio, porque detecta los datos excedentes del comienzo y permite descomprimir el archivo de todas maneras, simplemente ignorando el contenido de la imagen.
Si en cambio ésto fuera realizado intercambiando el orden de los archivos, por ejemplo, 'cat comprimido.zip imagen.png > imagenconcomprimido.zip', la imagen no podría ser visualizada por los motivos antes mencionados.

2.7. Ejercicio 6.7

El portador del video era el archivo 6a.wav.
El video encontrado era un fragmento en el que se explicaba una forma de ocultar información en telares usado en el pasado.

2.8. Ejercicio 6.8

El método de esteganografiado utilizado que no era LSB consistía simplemente en guardar los caracteres que formaban la frase que se quería guadar en formato hexadecimal en el archivo. En este caso en particular lo hacía en el final del mismo.
El método no es eficiente dado la información puede ser vista simplemente con un editor hexadecimal. Además de que el cambio que produce en la estructura del archivo original es muy grande ya que modifica muestras completas, en lugar del último bit de cada una de ellas.

2.9. Ejercicio 6.9

Al programa stegowav se le ha implementado una gran mejora, la cual consiste en permitir utilizar algoritmos LSBN, para $1 <= N <= 7$, dado que la implementación genérica de LSBN por así decirlo, resultaba mucho más prolija y con mínima repetición de código.
A futuro, se puede mejorar esta funcionalidad permitiendo N mayores, hasta el tamaño de la muestra, lo cual no representa una dificultad considerable.
Otra mejora útil podría ser permitir distintos tipos de archivos carrier, y no sólo .wav. Esto tampoco representa una gran dificultad.
Por último, podría implementarse una mejora que permita pedirle al programa que comprima la salida esteganografiada (con o sin encripción) en un archivo .zip, y que lo embeba en un archivo de imágen, por ejemplo .png, tal y como se dio con la imágen obtenida del archivo 6c.wav.