EPOCH

Esteganografía y Estegoanálisis

Criptografía y Seguridad – 1er Cuatrimestre 2016

Cavo, María Victoria	53202
Di Nucci, Nicolás Santiago	54091
Mounier, Agustín	54037
Rossi, Melisa Anabella	54265

Fecha de entrega 27 Junio de 2016

Índice

Introducción

El trabajo práctico especial consistió en dos etapas: esteganografía y estegoanálisis.

La primer etapa consiste en la generación de un programa capaz de embeber y extraer data de archivos RIFF wave.

La segunda etapa, se basaba en predecir con qué métodos se ocultó data, y lograr extraerla de manera correcta.

Esteganografía

Esta etapa se basó en la generación de un programa, codificado en c, que recibirá archivos .wav y brindará la posibilidad de embeber archivos con diferentes técnicas LSB. La información a embeber podrá ser previamente encriptada con diferentes algoritmos y modos de cifrado.

Input

Parámetros de entrada

`- embed`	Indica que se utilizará stegowav para ocultar información
`- extract`	Indica que se utilizará stegowav para extraer información oculta.
`- in`	Archivo que se va a ocultar
`- p`	Archivo wav portador
`- out`	Archivo wav salida
`- steg`	Algoritmo de esteganografiado (LSB1, LSB4, LSBE)
`- a`	Algoritmo de cifrado
`- m`	Modo utilizado en el cifrado
`- pass`	Contraseña del cifrado

Para que el programa funcione correctamente debe poseer el parámetro de entrada -embed o –extract al principio de la consulta.

Implementación

Stegowav fue implementado utilizando c como lenguaje de programación. Para poder utilizarlo bastara con realizar un make y ejecutar el archivo de salida stegowav.

Distribución de archivos

bytesmanager.c : En este archivo podemos encontrar las funciones que se encargan de conversiones entre arrays de bytes a su numero representado teniendo en consideración si estos se encuentran en big endian o little endian. Esto fue necesario para garantizar la portabilidad del programa.
ciphermanager.c : Posee todas las funciones encargadas de la encripción y desencripción de la data con diferentes algoritmos y modos.

lsb.c : Se encuentran todas las funciones que se encargan de embeber y extraer data con las diferentes técnicas propuestas por la cátedra.

stegowav.c : Esta es la clase principal que contiene el main. Podremos encontrar allí el pares del input del usuario.

wavmanger.c : En este archivo se encargar de parsear el wav y generar la estructura que se utilizará durante todo el programa.

Estructuras

CIPHERSTR : Utilizada para guardar los datos utilizados para el cifrado y descifrado del archivo a embeber/extraer.

```
typedef struct {
        unsigned char alg;          // algoritmo de cifrado
        unsigned char mode;             // modo de cifrado
        char* pass;                     // contraseña
} CIPHERSTR;
```

EMBEDSTR : Utilizada para guardar los datos necesarios para embeber un archivo.

```
typedef struct {
        unsigned char tech;             // técnica de esteganografía
        char* infile;               // archivo de entrada
        char* stegowav;                 // nombre de archivo de salida
        WAVSTR* wav;                    // archivo portador .wav
        CIPHERSTR* cipher;              // estructura de cifrado
} EMBEDSTR;
```

EXTRACTSTR : Utilizada para guardar los datos necesarios para embeber un archivo.

```
typedef struct {
        unsigned char tech;             // técnica de esteganografía
        char* outfile;              // archivo de salida
        WAVSTR* wav;                    // archivo portador .wav
        CIPHERSTR* cipher;              // estructura de cifrado
} EXTRACTSTR;
```

WAVSTR: Utilizada para el manejo de archivos wav.

```
typedef struct {
        RIFF_CK riff_desc;
        FMT_CK      fmt;
        DATA_CK data;
} WAVSTR;

typedef struct {
        BYTE  chunkID[4];
        CKSIZE      chunkSize;
        WORD        wFormatTag;
        WORD        wChannels;
        DWORD       dwSamplesPerSec;
        DWORD       dwAvgSamplePerSec;
        DWORD       dwAvgBytesPerSec;
        WORD        wBlockAlign;
        WORD        wBitsPerSample;
        WORD        extraParamSize;
        BYTE*       extraParams;
} FMT_CK;

typedef struct {
        BYTE    chunkID[4];
        CKSIZE  chunkSize;
        BYTE    format[4];
} RIFF_CK;

typedef struct {
        BYTE  chunkID[4];
        CKSIZE      chunkSize;
        BYTE*       soundData;
} DATA_CK;
```

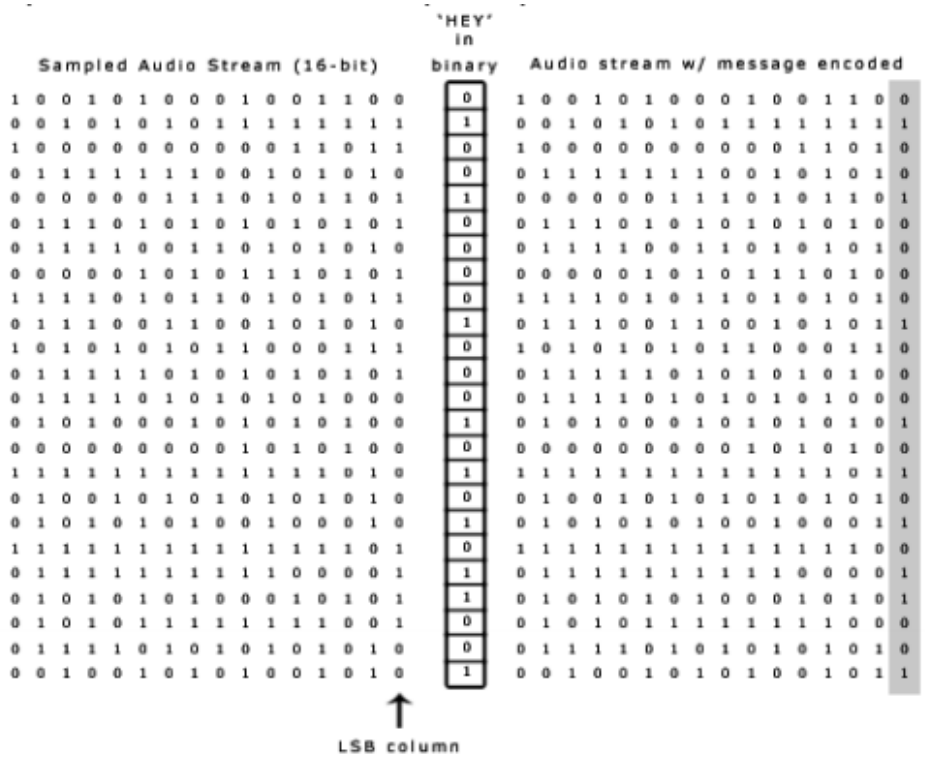
Técnicas de esteganografía

Para embeber archivos dentro de los .wav se implementaron 3 técnicas diferentes.

LSB1

Este método consiste en insertar la información substituyendo al bit menos significativo de cada byte del archivo portador, por un bit del archivo a embeber.

Para la implementación, sin embargo, se reemplaza el bit menos representativo por muestra. Es decir, en los casos en que el tamaño de la muestra es mayor a 1 byte, ocurre que no todos los bytes del archivo portador son modificados.



LSB4

Este método es similar a LSB1 con la diferencia que este reemplaza los últimos 4 bits. Esto permitirá embeber archivos de mayor tamaño en un mismo archivo portador pero generará mas ruido que LSB1 por lo que aumenta las posibilidades de notar que existe información embebida.

LSBE

Este método es similar a LSB1 con la diferencia de que escribe el último bit de aquellos bytes que posean 0xFE 0xFF.

Algoritmos y modos de cifrado

Stegowav permite embeber y extraer archivos con las siguientes características:

Algoritmo de cifrado

- → aes128
- → aes192
- → aes256
- → des

Modos de cifrado

- → ECB
- → CFB
- → OFB

Estegoanálisis

La cátedra proveyó cuatro archivos: mamamia13.wav, mamamia13a.wav, mandauna13a.wav y mandauna13b.wav, de los cuáles se sabe que tienen información embebida con alguna de las tres técnicas de esteganografía vistas, y sólo el contenido de una de estas fue cifrado previamente.

Análisis del tamaño de los archivos

Como primer aproximación, se decidió ver según cada técnica, cuántos bytes de data habrían escondidos y ver si los bytes disponibles para esconder información son suficientes.

Para saber cuántos bytes hay disponibles para las técnicas LSB1 y LSB4 sólo se debe contar los bytes de sound data (dato que se puede averiguar del data chunk del header) y dividirlo por la cantidad de bytes que tiene cada sample. En cambio, para LSBE, hay que preprocesar el sound data y contar cuantos bytes de la forma 0xFE o 0xFF hay.

Luego, se levantaran los primeros 4 bytes de data según la metodología de cada técnica LSB, dónde supuestamente estaría la longitud de bytes del archivo. La cantidad de bytes embebidos serían este número más cuatro, que provienen de los bytes que levantamos para saber la longitud.

En el caso de LSBE y LSB1, como esconden un bit por byte, por cada byte de data voy a requerir ocho. Así que la cantidad de bytes requeridos va a ser ocho veces la cantidad de bytes a embeber. En cambio, LSB4, como esconde cuatro bits por byte, por cada byte de data voy a requerir sólo dos. Por lo que la cantidad de bytes requeridos va a ser el doble de la cantidad de bytes a embeber. Si la cantidad de bytes disponible es menor que la requerida sabremos que no se pudo haber embebido información con esta técnica.

En *la tabla 1*, se muestra los resultados obtenidos de verificar esta información para cada archivo wav. Debido a la información provista por la cátedra, de que se utilizaron los tres métodos y que sólo uno contiene información cifrada deducimos la siguiente información:

- mandauna13b.wav tiene data embebida con LSBE, dado que es el único que indicó que sería posible.
- mamamia13.wav y mandauna13a.wav tienen data embebida con LSB4, y nos arriesgaríamos a suponer que la data de uno de los dos es la que está cifrada.

Por lo tanto, **la data embebida de mamamia13a.wav fue con la técnica LSB1.**

Archivo	Técnica	Bytes Disponibles	Bytes Requeridos	¿Es posible?
mamamia13.wav	LSB1	7682609	939530000	NO
	LSB4	7682609	8198	SI
	LSBE	1124355	4227857432	NO
mamamia13a.wav	LSB1	7682609	7264552	SI
	LSB4	7682609	7652	SI
	LSBE	1124355	1497625072	NO
manadauna13a.wav	LSB1	6724233	18521160	NO
	LSB4	6724233	87398	SI
	LSBE	211214	3514818584	NO
mandauna13b.wav	LSB1	6724233	327720	SI
	LSB4	6724233	469769700	NO
	LSBE	211214	51288	SI

Tabla 1 : Análisis del tamaño de los archivos según las técnicas LSB

Análisis de contenido

A través del análisis anterior, se pudieron descartar algunas técnicas para cada archivo.

Para hacer un análisis más profundo, indicando con que técnica se levanta la información, se levantó la data y se escribió en un archivo sin extensión que luego fue analizado para tratar de ver si se reconoce el tipo de archivo.

Para analizar los archivos se utilizó el comando file de Linux, el programa trID [1] .

Archivo	Técnica	File	trID
mamamia13.wav	LSB4	data	MacBinary
mamamia13a.wav	LSB1	data	Unknown
	LSB4	data	Unknown
manadauna13a.wav	LSB4	PNG	PNG
mandauna13b.wav	LSB1	PDP -11	Unknown
	LSBE	PDF	PDF

Tabla 2 : Análisis del tipo de los archivos

Como vemos en la *tabla 2* tanto File como trID pudieron distinguir la existencia de un archivo PNG en mandauna13a.wav y un archivo PDF en mandauna13b.wav.

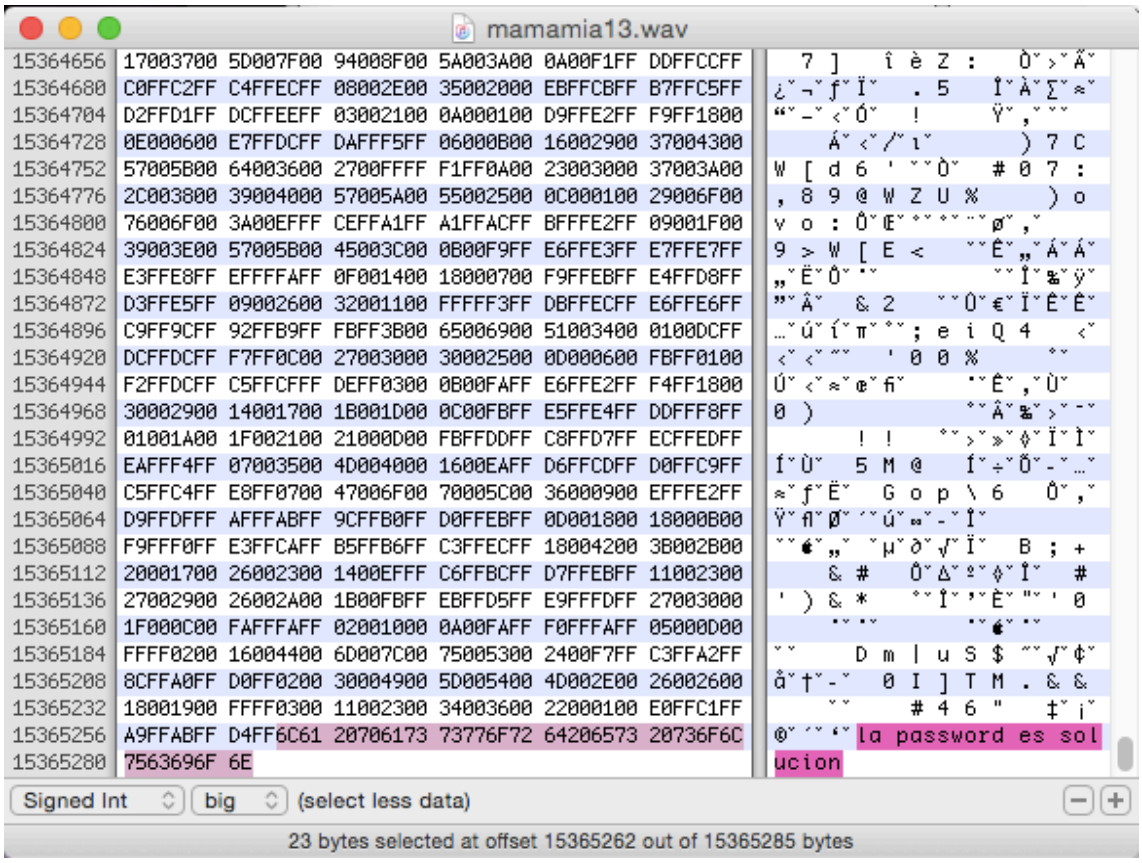
El próximo paso fue extraer el PNG y el PDF con el programa de estenografía implementado para la primer parte del trabajo. El PDF indica cambiar la extensión del archivo PNG a .ZIP y descomprimirlo, al hacerlo nos encontramos con una pista:

"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."

Esto nos permitió resolver el buscaminas del PNG, y como se muestra en la figura se haya que el algoritmo de encripción es aes256 modo cfb.

Figura 1: Imagen PNG con el buscaminas exraído del archivo mandauna13a.wav, buscaminas resuelto y resolución de la clave.

Por último para obtener la contraseña se utilizo una herramienta que permitió ver que en el archivo mamamia13.wav existía información oculta embebida con la técnica EOF la cual escribe el archivo portador luego de EOF por lo que no es procesada por muchos programas. Al analizar este archivo pudimos obtener que la contraseña era: SOLUCION.



Con esta información descartamos que el archivo que contiene el video a extraer es mamamia13a.wav y sabíamos que el algoritmo de cifrado era aes256 con el modo cfb8 cuya contraseña era solución.

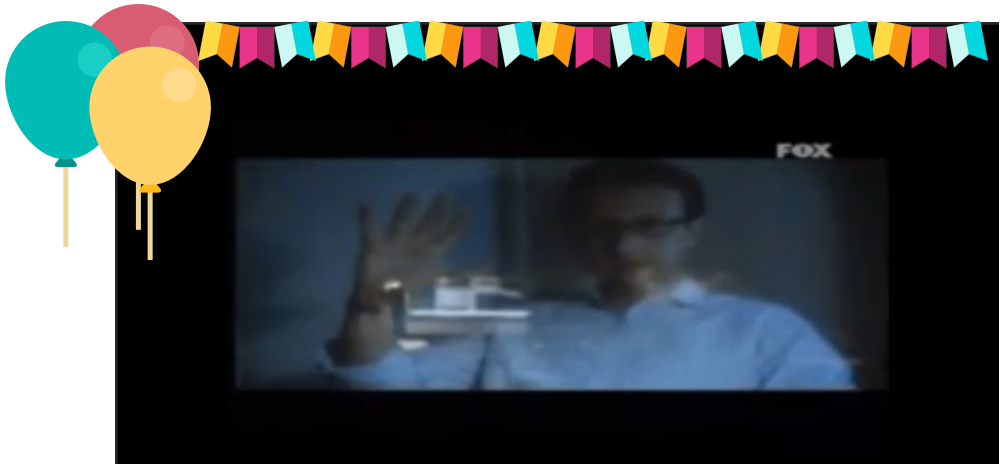
Con el análisis del tamaño para el archivo mamamia13a vimos que había dos técnicas de esteganografía posibles para embeber que eran LSB1 y LSB4. Teniendo en cuenta que el archivo a extraer era un video descartamos LSB1 por la poca cantidad de bits que permite embeber por lo que como resultado final se llego:

Archivo : mamamia13a.wav

Algoritmo de cifrado : aes256

Modo de cifrado: cfb8

Técnica de esteganografía: LSB4



Análisis cátedra

1. **Para la implementación del programa stegowav se pide que la ocultación comience en la primer muestra del archivo de audio. ¿Sería mejor empezar en otra ubicación? ¿Por qué?**

   De comenzar a esconder información en otra parte que no sea el primer sample, debería en algún lado fijo esconder donde comenzaría o comenzar siempre en el mismo lugar, otra opción seria empezar a escribir en el final del archivo de atrás para adelante. Esto podría concluir con que la información podría ser más difícil de encontrar, ya que la persona que busque la información secreta debería saber de qué manera estamos escondiendo.

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

Técnica	Ventajas	Desventajas
LSB1	-Distorsiona poco los archivos portadores	-Se requiere que el tamaño del archivo portador sea al menos 16 veces mayor que el que queremos esconder (para sample de 16 bits)
LSB4	-Puede guardar más información que las demás técnicas	-Distorsiona más el archivo portador
LSBE	-Distorsiona menos el archivos portador	-No se sabe cuánta información puede esconder a partir de el tamaño del archivo portador -Requiere un mayor procesamiento -Generalmente pueden esconder poca información en función del tamaño del archivo portador

3. **Para la implementación del programa stegowav 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?**

   No sería conveniente debido a que, además de no saber donde arranca la data, estarías delatando que el archivo no es cifrado y habría ambigüedad a la hora de leer los archivos, porque primero intentarías leer una extensión cuando en realidad no la hay. Por otro lado, esta extensión brindaría información al atacante

haciendo que su trabajo sea más simple para aquellos archivos que poseen un header conocido.

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

Este punto se explico previamente en Análisis de tamaño de archivos y Análisis de contenido.

5. **¿Qué se encontró en cada archivo?**

Archivo	Técnica	¿Qué se encontró?
mamamia13.wav	LSB4	Password
mamamia13a.wav	LSB1	Archivo de video
mandauna13a.wav	LSB4	Imagen PNG
mandauna13b.wav	LSBE	Archivo PDF

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

El png contenía adentro un archivo .zip que fue embebido con la técnica EOF. Esto es posible debido a que la lectura de los archivos .zip es de abajo hacia arriba al revés de los archivos PNG (mutuamente embebidos).

7. **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 del video era mamamia13a.wav.

8. **¿De qué se trató el método de estenografiado que no era LSB? ¿Es un método eficaz? ¿Porqué?**

El método EOF, utilizado en el archivo mamamia13b.wav, se basa en ocultar información luego del EOF del archivo. Algunos archivos se leen teniendo en cuenta el tamaño especificado en su header, de manera tal que asumen que no hay más información una vez terminado el mismo. Esta técnica de ocultamiento es muy fácil de detectar ya que basta con comparar el tamaño que indica el header del archivo con el su tamaño real.

9. ¿Qué mejoras o futuras extensiones harías al programa stegowav?

Como futuras extensiones del programa stegowav se podría considerar agregar soporte para otros archivos de audio como mp3 y flac, así como también otros tipos de archivos como imágenes o ejecutables. Por otro lado, se podrían agregar otros tipos de algoritmos de esteganografía como “Echo Hiding” el cual oculta información generando un pequeño eco en el audio o “Hiding in silence intervals” el cual oculta la información en los silencios de un discurso hablado.

Además se podría agregar la posibilidad de soportar todos los tipos de wav que hay y no solo PCM.

Referencias

- [1] trID – File Identifier http://mark0.net/soft-trid-e.html
- http://soundfile.sapp.org/doc/WaveFormat/
- http://www.elladodelmal.com/2008/01/mini-tutorial-de-esteganografaestegoanl.html
- http://www.jatit.org/volumes/research-papers/Vol5No6/15Vol5No6.pdf