2022

Integrantes:

57735 - Josefina Assaff Alvarez

60567- Justina Vacas Castro

60584- Ian Mejalelaty

Grupo 22

Fecha de entrega: 25/06

Informe Trabajo Práctico

72.44 – Criptografía y Seguridad

1. Discutir los siguientes aspectos relativos al documento.

a. Organización formal del documento.

El documento está bien seccionado, con una muy clara introducción del tema y ejemplos básicos que aportan mucho al entendimiento del contexto. Lo único por ahí medio raro era el tema de las referencias a imágenes y textos del anexo que no resultaban muy útiles en la mayoría de los casos (lo único que nos sirvió fueron los tableros). En la parte de los algoritmos propuestos por ahí hubiese estado bueno poner más ejemplos con imágenes o con secuencias de bits de una manera más clara como lo hicieron al explicar el método LSB simple.

b. La descripción del algoritmo.

El algoritmo nos costó un poco entenderlo al principio ya que el ejemplo que proponen no es muy claro y no acompañan la explicación con alguna imagen o colores que permitan distinguir bien los bits fundamentales a tener en cuenta a la hora de implementar el algoritmo. Más allá de eso, la descripción es concisa y explica bien claro y resumido el objetivo del mismo. Las tablas de ejemplo en el anexo con los contadores y los bits cambiados también ayudan mucho a la comprensión del mismo. El algoritmo en sí es bastante simple de implementar y muestra buenos resultados en algunos casos (sobre todo el segundo tipo).

c. La notación utilizada, ¿es clara? ¿Hay algún error o contradicción?

Es un poco confusa la explicación con los “2nd y 3rd LSB” la primera vez que se lee ya que el ejemplo no lo aclara muy bien y uno puede confundirse fácilmente con el orden de los bits. Pero por todo lo demás está bastante claro, no notamos ningún error.

2. 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.

|  |  |  |
| --- | --- | --- |
| Técnicas | Ventajas | Desventajas |
| LSB1 | * Fácil de implementar * Poca alteración del archivo original | * Poca capacidad de ocultamiento |
| LSB4 | * Algoritmo más rápido * Mayor capacidad de ocultamiento * Fácil de implementar | * Distorsiona mucho el archivo portador * Muy fácil de detectar mediante entropía |
| LSBI | * Más difícil de detectar mediante entropía * Menos cantidad de bits modificados que LSB1 (archivo más puro) | * Difícil de implementar * Tarda más * Ocupa 4 bytes extra * Poca capacidad de ocultamiento |

A continuación compararemos mediante entropía (usando el comando de binwalk -E) que tanto se ve afectado el archivo original en sus distintas versiones de esteganografiado:

Chart, line chart, histogram

Description automatically generatedChart, line chart

Description automatically generated

ORIGINAL LSB1

Chart, line chart

Description automatically generatedChart, line chart, histogram

Description automatically generated

        LSB4 LSBI

Podemos ver que se ve un cambio muy grande al principio en el método LSB4 comparado con el original, lo que indica un desorden en el nivel de información de esa zona y podría hacernos sospechar la presencia de un archivo oculto.

3. 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.

Cada archivo .bmp lo probamos con los distintos métodos de estenografía: LSB1, LSBI, LSB4 sin encriptado. Estos fueron los resultados exitosos:

Con LSB1

Con el archivo roma.bmp → encontramos un .png que mostraba un buscaminas resuelto



Con LSBI

El archivo bogota.bmp → encontramos un pdf

Graphical user interface, text, application, email

Description automatically generated

Seguimos los pasos del archivo pdf y al descomprimir el .zip encontramos un archivo sol22.txt el cual indicaba los pasos a seguir para descifrar el mensaje de la primera imagen encontrada del buscaminas. Gracias a ellos pudimos encontrar el modo de encriptación “AES 192 ECB ”.

Text

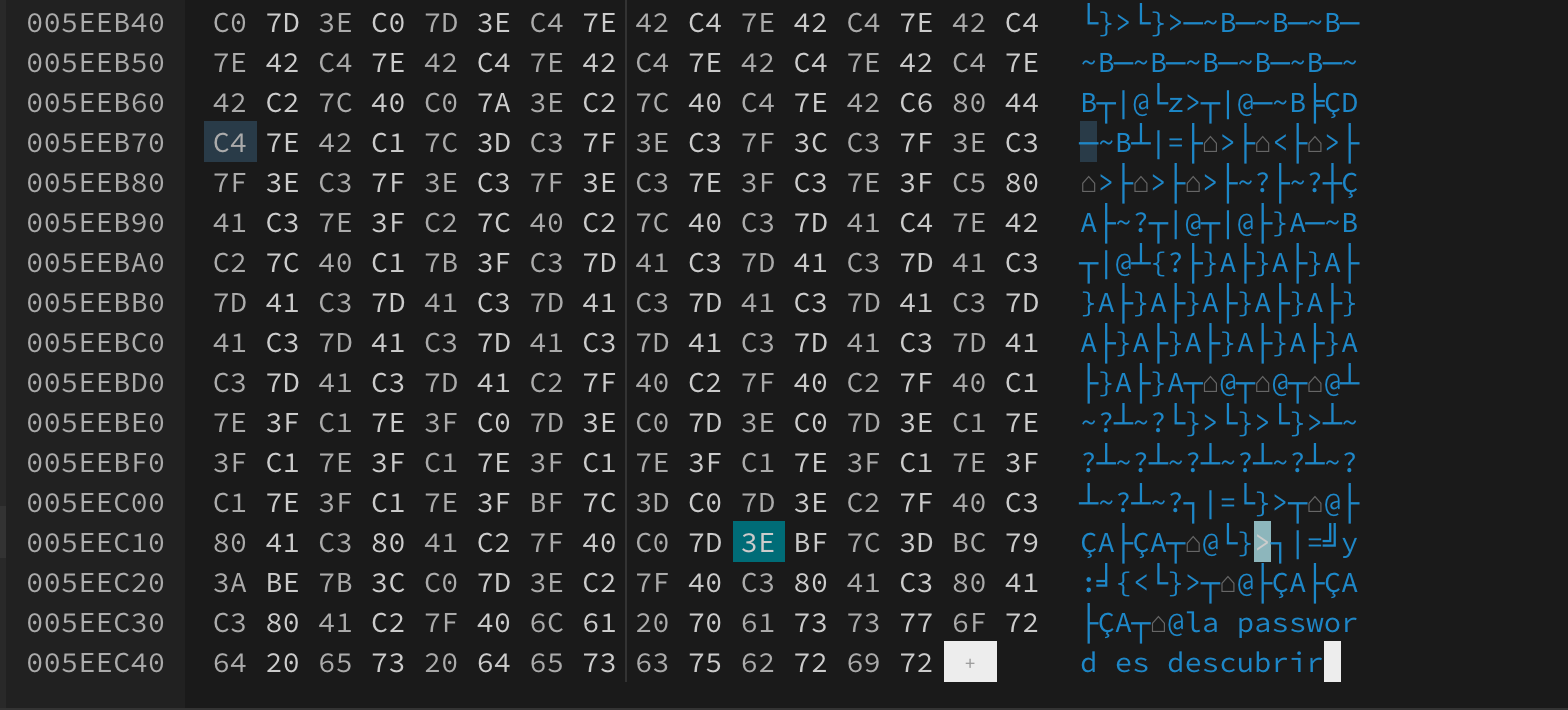
Description automatically generated

A continuación la imagen del buscaminas teniendo en cuenta los pasos del archivo “sol22.txt”

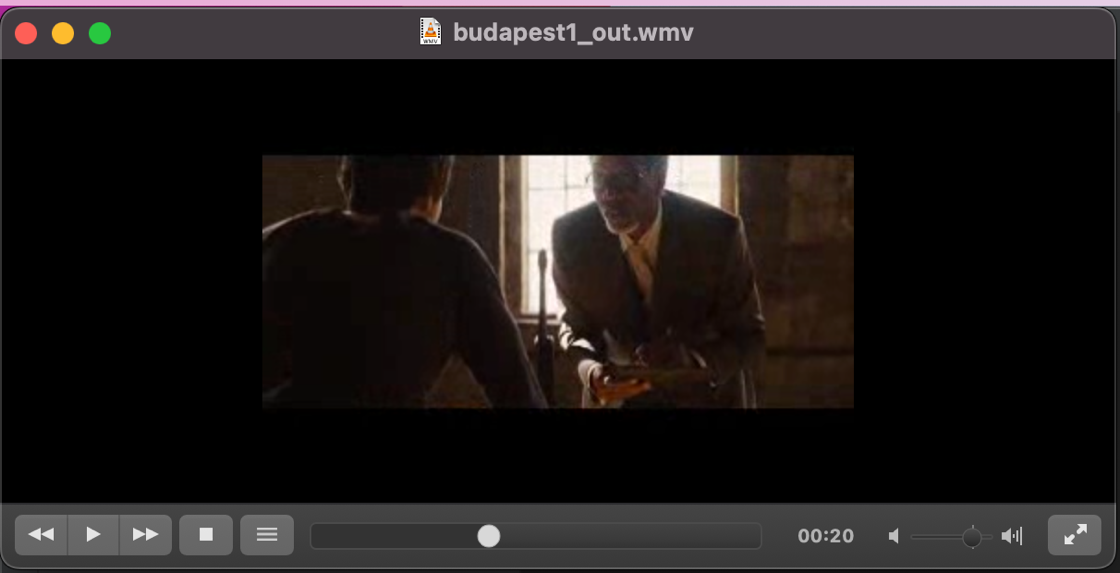
Application

Description automatically generated with low confidence

Luego abriendo con un editor hexadecimal todos los archivos, en budapest.bmp encontramos al final el mensaje que indica la contraseña “descubrir”.



Por último en el archivo budapest1.bmp usando LSB4 y el método de desencriptación AES-192 ECB pudimos sacar el archivo .wmv siguiente:



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

El archivo .png que obtuvimos del archivo roma.bmp contenía 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 contrario de los archivos png, mutuamente embebidos.

5. Uno de los archivos ocultos era una porción de un video, donde se ve ejemplificado una manera de ocultar información ¿qué se ocultaba según el video y sobre qué portador?

En el video nos explican que una secta descubrió una manera de esconder mensajes binarios en tejidos. Si un hilo pasaba por encima de otro hilo, era un 1, y si pasaba por debajo, era un 0. Según el video el mensaje oculto era “un nombre, un objetivo”, y su portador era un tejido.

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

El método se trata de escribir al final del archivo. Esto no es eficaz ya que solo basta con abrir el archivo con un editor de texto o hexadecimal y encontrar al final el mensaje oculto. Un control del header permite ver que algo no anda bien en el archivo.

7. ¿Por qué la propuesta del documento de Akhtar, Khan y Johri es realmente una mejora respecto de LSB común?

La propuesta del documento es una mejora ya que nos muestran empíricamente como el número de bits cambiados de la imagen original se reduce significativamente (sobre todo con el segundo método), lo que permite obtener un resultado mucho más discreto y difícil de detectar.

8. ¿De qué otra manera o en qué otro lugar podría guardarse el registro de los patrones invertidos?

Se podrían poner dentro de los primeros 4 bytes (o cualquier otros 4 en realidad) pero usando los cuartos LSB y dejando los primero 3 LSB para el esteganografiado LSBI.

Ejemplo: 11011010 00101100 11000101 10101010

El registro sería 00 = 1, 01 = 1, 10 = 0, 11 = 1. Y los LSB se usarían para el esteganografiado del tamaño.

Otra posibilidad, si el texto está encriptado, podría ser pasarlo junto a la contraseña (por ahí este método sería un poco menos seguro). Ejemplo: “descrubrir1010”

9. Leer el Segundo esquema y analizar (sin implementar) cuáles serían las ventajas que pueden verse.

El segundo esquema, si bien es más tedioso de implementar, la precisión que otorga el escaneo dentro de los Bit pattern (3rd,  2nd LSB) disminuye drásticamente la cantidad de bits cambiados de la imagen original, lo cual nos otorga un ocultamiento mucho más sutil y difícil de detectar.

10. Leer el Segundo esquema e indicar qué desventajas o inconvenientes podría tener su implementación.

Si bien los resultados de este método son muy buenos, no justifican que la implementación sea tan costosa en tiempo y complejidad. Lo más tedioso que notamos en esta técnica es el hecho de que el decoder deba tener tanto la imagen esteganografiada como la original.

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

Para la parte de decoding no tuvimos problemas ya que rápidamente pudimos implementar una función para obtener los registros de bits invertidos y guardarlos en un array como si fuesen flags y a partir de ahí simplemente era copiar el método de LSB1 pero ir chequeando los 3rd,  2nd LSB para ver si coincidían o no con uno de los patrones que tenían flag encendido y en ese caso invertir el LSB.

Luego para la parte de encoding nos costó un poco entender que debíamos hacer todo en dos etapas: primero realizar un escaneo auxiliar de todo para poder contar los bits que se iban a modificar o no y setear los patrones a invertir, y luego recién ahí hacer el esteganografiado definitivo con los bits invertidos. Para eso utilizamos dos funciones de LSBI, una de escaneo que solo lee y cuenta, y luego una de encoding que lee de nuevo, invierte los bits necesarios y luego los escribe en el archivo de salida. También tuvimos algunos bugs por momentos con el tema de mantener tracking de la posición de los punteros de lectura y escritura de los archivos en la etapa de encoding.

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

Como mejora creemos que podríamos hacer que el programa te sugiera con qué método de esteganografiado fue ocultado el archivo en el proceso de decoding en caso de detectar que el método ingresado está mal. Podríamos chequear el tamaño del archivo oculto que está al principio con cada uno de los métodos y casi siempre el método correcto es el único que devuelve un tamaño menor al tamaño del archivo portador.

También podríamos hacer que en el proceso de encriptado te genere una clave aleatoria automáticamente (si no ingresaste ninguna por comando) y que te devuelva un .txt con la clave generada para compartirla con el receptor.

Y ya pensando a largo plazo podríamos intentar hacer que el programa solo te pida la contraseña y detecte automáticamente el método de desencriptado a utilizar ya sea por prueba y error o por detección mediante el escaneo de la data desesteganografiada.