Trabajo Práctico de Implementación: Secreto Compartido con Esteganografía

Grupo 6

Integrantes:

- Méndez, Ignacio Alberto
- Roca, Agustín
- Barbieri, Guido

Introducción

El objetivo de este trabajo práctico es aprender sobre el concepto de Esquema de Secreto Compartido (ESC), y aplicarlo en el área de la esteganografía LSB, para así lograr un Esquema de Imagen Secreta Compartida (EISC).

En particular, se trabajó sobre el ESC de Shamir con un umbral (k,n), en el cual la información secreta S se comparte entre n participantes, y se necesitan mínimo k de esos participantes para recuperar el secreto (entonces se deduce que $k \leq n$). La carga máxima de datos útil puede ser controlada mediante el valor k. Esto es, si k es mayor que 4, la máxima carga útil de los datos secretos puede ser tan grande como el tamaño de la imagen camuflaje, lo cual permite una eficiente y secreta transmisión de los datos.

Para la implementación del EISC, nos basamos en el documento entregado por la cátedra. Este habla sobre el uso de los polinomios de Lagrange en el campo de Galois GF(2^8) para codificar y decodificar las imágenes secretas sin pérdidas.

Codificación

La codificación consiste en compartir los datos secretos utilizando el esquema de secreto compartido de Shamir bajo un campo GF(2^8) y luego cada sombra es insertada dentro de n imágenes camuflaje usando esteganografía LSB.

La razón por la que usamos las operaciones polinomiales en el campo GF(2^8) es porque contribuyen a la recuperación sin pérdida de los datos (en este caso bytes). Esto permite trabajar no solo con imágenes, sino también con archivos ejecutables e incluso con datos encriptados. En este aspecto, las operaciones polinomiales en el campo de Galois provisto son superiores a las operaciones usando congruencias módulo.

En cuanto a la elección de k (el número mínimo de sombras necesarias para recuperar el secreto), esta está directamente relacionada con la calidad final de las imágenes portadoras. Cuanto más alto es k, en menos partes se dividirá el secreto pero utilizando polinomios de grado mayores y más imágenes portadoras. Esto haría que se modifiquen menos píxeles de las imágenes portadoras.

Una vez que se eligió el k, se procede a seleccionar únicamente los datos de la imagen secreta (es decir, saltándonos su encabezado) y dividirlos en píxeles que se repartirán en las imágenes portadoras.

Es importante aclarar que es posible dividir estos datos en bytes en vez de píxeles. Esto nos hubiera permitido guardar no solo los datos de la imagen secreta sino también su encabezado en las imágenes portadoras. Sin embargo, habría que contar con alguna técnica para agregar bytes al final en caso de que la división no haya sido divisible por k.

Para probar la codificación en este TPE, véase el README ubicado en la carpeta raíz del proyecto.

Decodificación

Como estamos en un contexto de Esquema de Secreto Compartido con un umbral (k,n), se necesitan por lo menos k imágenes portadoras para poder recuperar la imagen secreta.

Se recalca la importancia de haber utilizado operaciones polinomiales en el campo GF(2^8) tanto en la codificación como en la decodificación, ya que gracias a esto es que no se pierden datos en la recuperación del secreto.

Para probar la decodificación en este TPE, véase el README ubicado en la carpeta raíz del proyecto.

Resultados

A continuación, se mostrarán los resultados obtenidos en nuestro TPE.

Codificación

Primero, mostraremos la imagen secreta que se usó en la distribución. Esta se obtuvo del Campus, provista por los profesores.

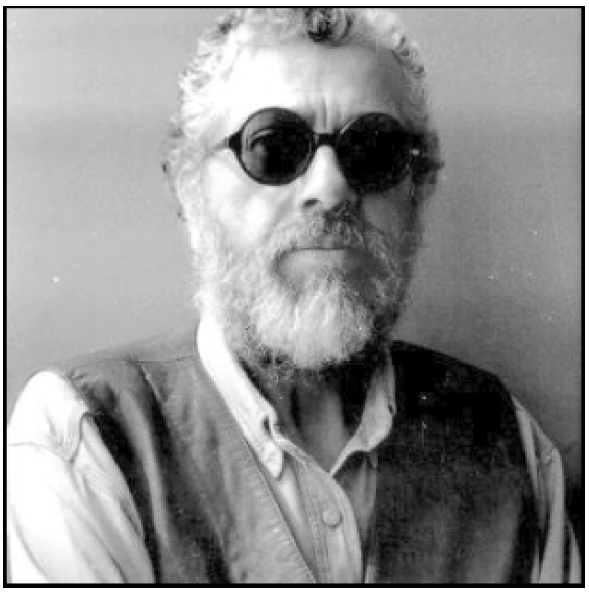

Imagen secreta

A continuación, se mostrarán las imágenes portadoras antes de la distribución. Es decir, no estarán portando ningún secreto. Estas se obtuvieron del Campus, provistas por los profesores.

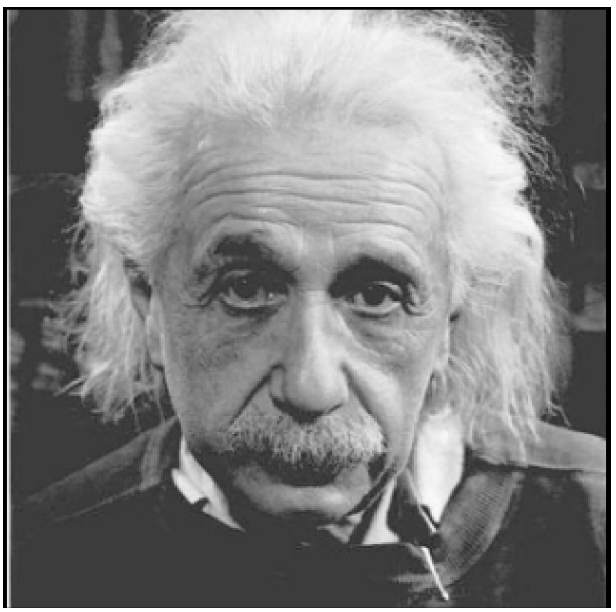

Imagen portadora 1 antes de la codificación

Imagen portadora 2 antes de la codificación

Imagen portadora 3 antes de la codificación

Imagen portadora 4 antes de la codificación

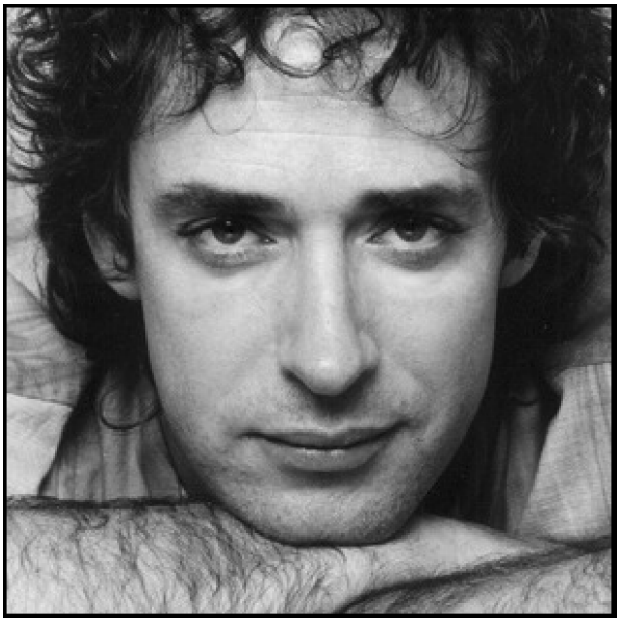

Imagen portadora 5 antes de la codificación

Imagen portadora 6 antes de la codificación

Ahora, mostraremos las mismas imágenes portadoras, pero luego de la distribución. Es decir, ahora sí están portando el secreto. Estas imágenes no fueron provistas por los profesores, sino que fueron generadas por nuestra implementación.

El k que se usó fue 6.

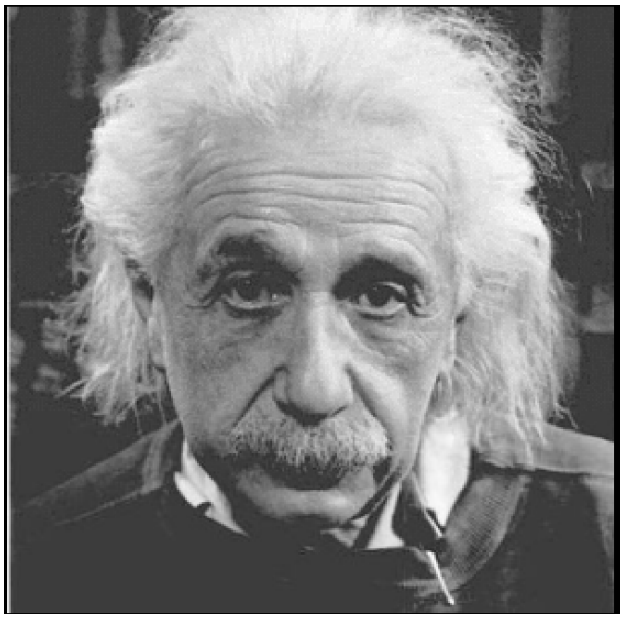

Imagen portadora 1 después de la codificación

Imagen portadora 2 después de la codificación

Imagen portadora 3 después de la codificación

Imagen portadora 4 después de la codificación

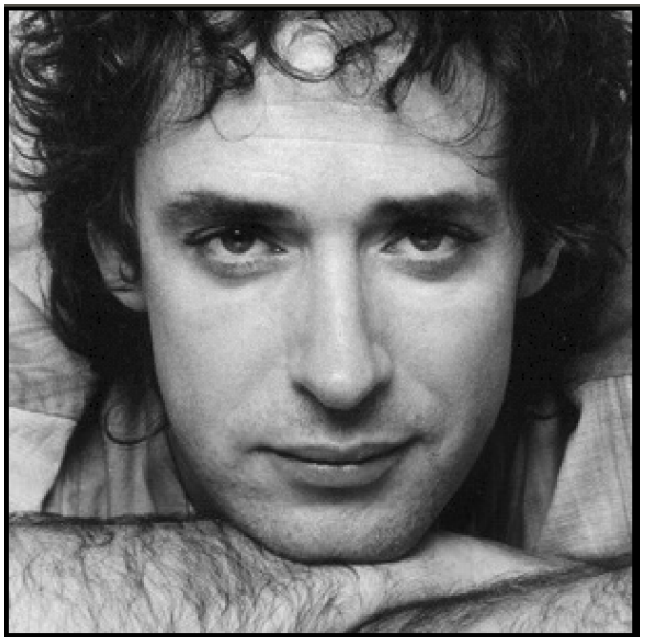

Imagen portadora 5 después de la codificación

Imagen portadora 6 después de la codificación

Si se observa cuidadosamente, por ejemplo la imagen portadora N°2, luego de la codificación se ve que la sección negra de la parte de arriba se encuentra un poco pixelada. Este fenómeno ocurre en las secciones negras de las imágenes. Sin embargo, la calidad general de la imagen no se vio deteriorada, y a fines de este TPE, creemos que cumple con el objetivo.

Decodificación

Primero, mostraremos las imágenes portadoras que tienen guardado parte del secreto adentro. Estas se obtuvieron por mail, provistas por los profesores.

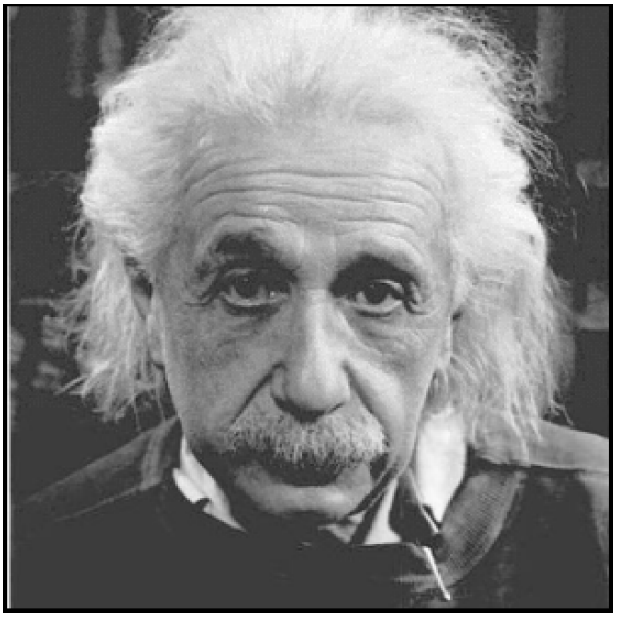

Imagen portadora 1 antes de la decodificación

Imagen portadora 2 antes de la decodificación

Imagen portadora 3 antes de la decodificación

Imagen portadora 4 antes de la decodificación

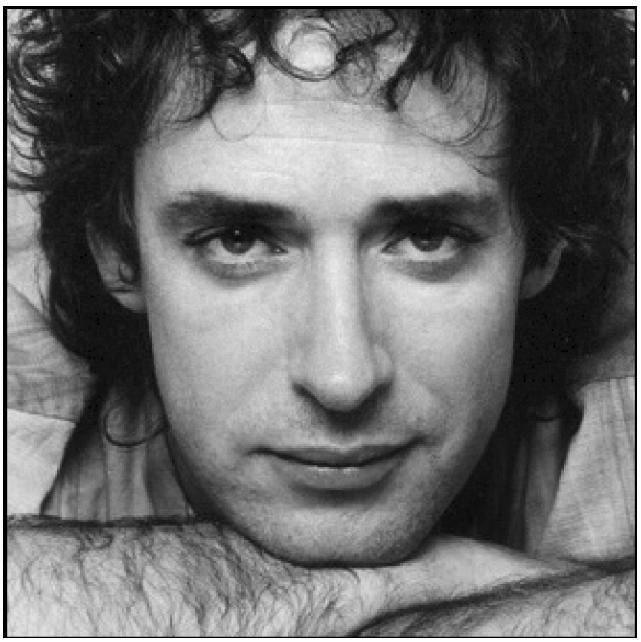

Imagen portadora 5 antes de la decodificación

Imagen portadora 6 antes de la decodificación

A continuación, mostraremos la imagen secreta que se recuperó de las anteriores 6 imágenes portadoras. Es importante notar que se tuvieron que usar las 6 imágenes para recuperar el secreto, con menos de 6 no funcionó. Esto indica que el secreto se dividió en 6; o sea que k fue 6.

Secreto recuperado

Como se puede ver, el secreto fue recuperado exitosamente.

Observaciones

1. El documento provisto por la cátedra fue de gran ayuda para el entendimiento de los conceptos y detalles de la implementación. Sin embargo, nos fue difícil entenderlo al comienzo, en particular la sección de interpolación de Lagrange, en donde no nos quedaba claro qué había que iterar.
   Por otra parte, el documento abusa de la palabra “bloque” usándola para los bloques de la imagen secreta pero también para los bloques 2x2, complicando el entendimiento general de los conceptos.

2. El algoritmo implementado en este TPE fue fácil de hacer una vez entendidos los conceptos. No son muchas las *features* que hay que desarrollar, y una vez hecha la codificación, la decodificación es conceptualmente muy simple ya que es hacer la operación inversa.

3. El polinomio generador usado en este TPE podría cambiarse. Pero es necesario que todas las partes sepan de este cambio. Es posible usar al polinomio generador como clave.

4. Este TPE **tiene una vulnerabilidad.** Si una de las imágenes portadoras tiene todos sus píxeles en negro, se puede recuperar una k-ésima parte de la imagen original decodificando solamente esa imagen portadora, que a k más chicos especialmente, podría deducirse la imagen igualmente. Esto sucede porque $f(0) = s_1$ sin importar el polinomio.
   A continuación, se muestra un ejemplo que probamos con nuestro TPE:

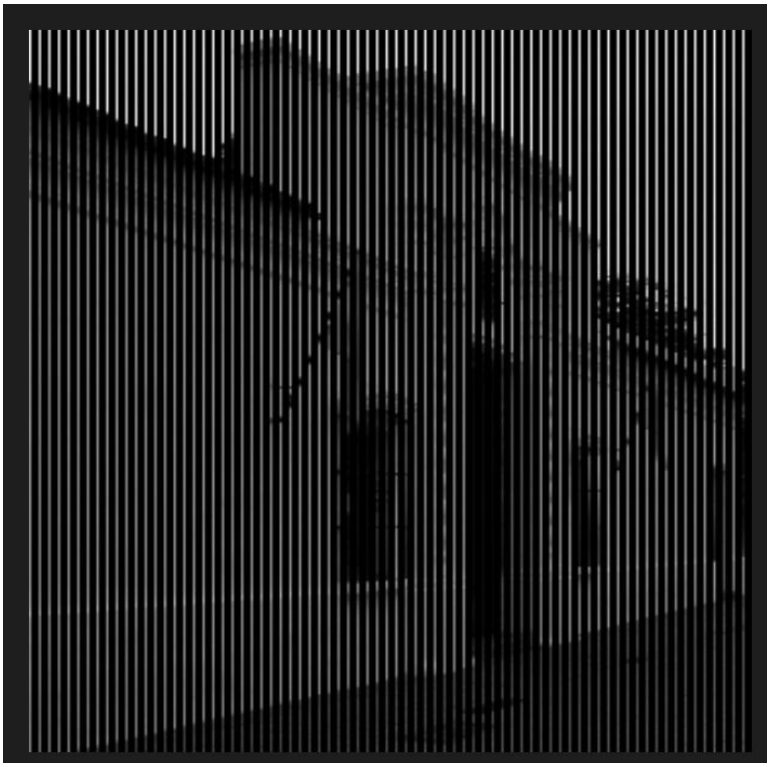

Imagen recuperada a partir de una sola imagen portadora negra (K=4)

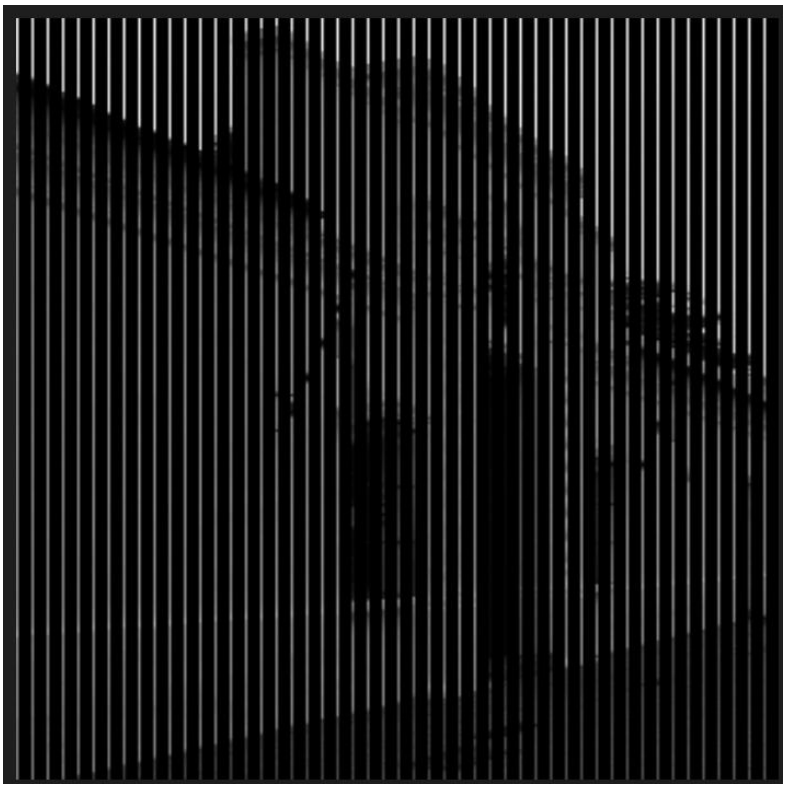

Imagen recuperada a partir de una imagen portadora negra (K=6)

Este es el caso más extremo en que todos los píxeles de la imagen portadora sean 0, pero en realidad con que solamente el bloque superior izquierdo de cada bloque de 2x2 esté en 0 es suficiente para obtener el mismo resultado.

Posibles extensiones

1. Es posible implementar este TPE usando imágenes a color (24 bits por píxel). Para hacerlo, habría que dividir los datos de la imagen secreta en bytes en vez de en píxeles. Lo que esto causa es que en vez de dividir la imagen secreta de a k píxeles, ahora habría que dividirla de a k secciones de RGB. Debido a esto, cada bloque dividido pasa a tener k/3 píxeles en vez de k, lo que a la vez causa que haya que ocultar el triple de datos en cada imagen portadora. Se debe tener cuidado con el tamaño de las imágenes portadoras si se decide usar imágenes a color.
2. En lugar de tomar los bloques como matrices 2x2, es posible usar cualquier otra convención. Sin embargo, es importante que esta nueva forma no produzca que en las imágenes portadoras se empiece a notar que fueron alteradas.

Conclusión

Observando los resultados obtenidos, se concluye que hemos cumplido con el objetivo de este TPE. Las imágenes portadoras se ven prácticamente idénticas al ojo humano cuando se les agrega su parte del secreto. Hemos aprendido que el ESC de Shamir con un umbral (k,n) es una alternativa ideal para sistemas cuya seguridad reside en que se necesita obligatoriamente al menos k integrantes para revelar el secreto. Esto causa que ningún participante tenga más “poder” que otro si el esquema está bien armado.