Trabajo Práctico
Criptografía y Seguridad

Secreto Compartido con Esteganografía

Integrantes

Baiges, Matías 59076
Comerci, Nicolás 59520
Piñeiro, Eugenia 59449

Criptografía y Seguridad

Secreto Compartido con Esteganografía

1. Recuperación del Secreto

El secreto se recuperó mediante la utilización de la interpolación de Lagrange a partir de 4 imágenes.

Sombras provistas por la cátedra:

El secreto recuperado fue el siguiente:

2. Implementación

En la implementación se permite modificar el polinomio generador *POLY_GEN*, y lo primero que se hace al invocar al ejecutable, es generar una tabla de multiplicación de Galois, al igual que una tabla de inversos multiplicativos, utilizando ese polinomio.

Tras invocar al programa como es señalado en el README, el programa carga las imágenes correspondientes (sea para distribuir un secreto o para recuperarlo), luego efectúa las operaciones de evaluación de los pixeles X (correspondientes a los pixeles XWVU) y

almacena P(X), donde P contiene coeficientes que corresponden con píxeles de la imagen original si se trata de distribuir el secreto, o bien recupera estos coeficientes a través de interpolación en caso de recuperar el secreto.

Para recuperar el secreto, se utiliza el método de interpolación de Lagrange propuesto en el documento, que de manera iterativa permite obtener los coeficientes del polinomio, sin manejo de productos entre polinomios.

Mejoras en la implementación

Durante la recuperación del secreto mientras se utilizan las k sombras necesarias para recuperar los píxeles de la imagen original, si alguna de ellas no verifica con el bit de paridad (checksum), se utiliza otra de las sombras en el directorio (si las hay) o completa con un color uniforme en caso de que no se pueda revelar ese conjunto de píxeles por error en checksum.

Para ver esto, utilizamos una imagen sombra donde se almacenó previamente el secreto (imagen de Marilyn Monroe), alterando algunos de los bytes en la imagen (cambiando píxeles por 0x00), de manera que algunos de ellos tendrían un checksum que no es válido.

El resultado obtenido al recuperar el secreto utilizando esta sombra alterada, fue:

Sin embargo, si se encuentra otra imagen correspondiente al esquema (k,n) que también permite recuperar este secreto, entonces a la hora de recuperar los bloques “corruptos”, si identifica que el checksum no valida, entonces utiliza esta otra imagen “sana” y permite obtener mejores resultados.

A su vez, se realizaron tests unitarios para probar la conversión de la matriz de píxeles a las matrices 2x2 de XWVU y viceversa, la correcta construcción de la matriz de inversos multiplicativos y diversas operaciones en campos de Galois.

3. Cuestiones a Analizar

1. Discutir los siguientes aspectos relativos al documento.

a. Organización formal del documento (¿es adecuada? ¿es confusa?)

b. La descripción del algoritmo de distribución y la del algoritmo de recuperación. (¿es clara? ¿es confusa? ¿es detallada? ¿es completa?)

c. La notación utilizada, ¿es clara? ¿cambia a lo largo del documento? ¿hay algún error?

La organización formal del documento es adecuada. La descripción del algoritmo de distribución y la del algoritmo de recuperación son claras aunque complejas, siendo de mucha utilidad los diagramas que se presentan a lo largo de todo el documento.

En cuanto a la notación utilizada, quizás es algo confuso seguir el hilo de la explicación con tantos símbolos. Por ejemplo, las letras *i,j* se utilizan como subíndices tanto como supraíndices, a veces encerradas con paréntesis y otra veces no. Luego, se agregan el ñuflo y el apóstrofe dificultando aún más la lectura:

$$U_{i,j}, u_7^{(i,j)}, u_8^{(i,j)} \quad p' = \tilde{u}_6^{i,j} \quad T'_{i,j} = [\tilde{w}_6^{i,j},$$

Además los guiones (-) no se muestran a lo largo de todo el documento lo cual en algunos casos hace que sea difícil entender las operaciones. Por ejemplo:

$$s_r^j = \left[(-1)^k \, r \left(\sum_{i=1}^{k} Y'_{i,j} \prod_{q=1, i \neq q}^{k} \frac{X_{q,j}}{X_{i,j} - X_{q,j}} \right) \right] \bmod g(x), \qquad Y'_{i,j} = \left[\frac{Y_{i,j} - s_1^j}{X_{i,j}} \right] \bmod g(x)$$

En la fórmula (7) nos encontramos que señala que para calcular Y', se realiza la resta entre Y_i,j y s_1, y luego se divide por X_i,j. Esto solo ocurre en la primera iteración (para calcular s2). La fórmula generalizada utiliza en verdad la resta entre Y'_i,j anterior y s anterior (en el caso de la iteración 3, s_2) para luego dividir por X_i,j.

En la fórmula (9) nos encontramos con que se encuentra el valor reemplazado de k en la fórmula, en vez de k.

Antes de esta fórmula señala:

"La carga útil (payload) de los datos usados en el algoritmo con un esquema de umbral (k,n) está dado por:"

$$Payload\ (byte) = \frac{NPixeles \times Ncolor \times 4}{4}$$

Efectivamente, la fórmula para un esquema (k, n) sería:

$$Carga\ útil\ =\ (N\ píxeles\ *\ N\ colores\ *\ k\ )\ /\ 4$$

2. El título del documento hace referencia a que optimiza la carga útil ¿a qué se refiere? ¿Qué relación existe entre k y el tamaño de la portadora?

La carga útil se trataría del máximo tamaño del secreto que el esquema de secreto compartido puede distribuir:

$$Carga\ útil\ =\ (N\ píxeles\ *\ N\ colores\ *\ k\ )\ /\ 4$$

siendo N píxeles el número de píxeles que posee la imágen, N colores el número de colores que tienen las imágenes camuflaje, k la variable del esquema de secreto compartido de umbral (k,n) y 4 la cantidad de píxeles que ocupa un XWVU (la matriz 2x2 con la cual se distribuye el secreto).

Como el documento lo señala, cuanto mayor k, mayor es la carga útil.

En cuanto a la optimización de la carga útil, se podría decir que hace referencia a esconder la información de manera tal que la imagen camuflaje tenga la mejor calidad posible. Esto se debe en parte a que los bits alterados son los 3 menos significativos dentro de los bloques XWVU, afectando a los bytes W,V y U, y en parte a la utilización de campos de Galois para almacenar el secreto.

La cantidad de píxeles de la imagen portadora debe ser divisible por K, para poder dividir todos estos píxeles en bloques adecuadamente. La cantidad de bloques que se verán afectados en las sombras será de la cantidad de pixeles en la imagen secreto dividido por K, de manera que la cantidad de píxeles afectados en las imágenes sombra será al menos de (#pixelesSecreto / K)*(4-1) (en el caso en el que solo se actualicen w,v,u).

3. ¿Qué ventajas y qué desventajas ofrece trabajar en GF(2^8) respecto de trabajar con congruencias módulo?

Al trabajar con congruencias módulo se puede obtener una imagen revelada con pérdidas. En cambio, utilizar GF(2^8) garantiza no solo que la imagen obtenida sea sin pérdidas sino también que las imágenes camuflaje posean mayor calidad.

Las desventajas de usar GF(2^8) es tal vez la complejidad de cálculo que llevan estas operaciones. En el caso de nuestra implementación sacrificamos espacio a costa de performance, puesto que almacenamos una tabla para multiplicar y una tabla para calcular los inversos multiplicativos.

4. ¿Se puede trabajar con otro polinomio generador? ¿podría guardarse como "clave"?

Se puede trabajar con otro polinomio generador mientras sea irreducible, de grado 8, con número impar de términos y al menos una constante. Por ejemplo, en la implementación de este proyecto se utiliza este: $p(x) = x^8+x^6+x^5+x+1$ con su representación en hexadecimal 0x163. Pero también se podría utilizar el siguiente: $p(x) = x^8+x^7+x^6+x^5+x^2+x+1$ o $p(x) = x^8+x^7+x^6+x^3+x^2+x+1$

En la implementación del proyecto se podría cambiar la constante *POLY_GEN* que representa los coeficientes del polinomio en hexa en *main.c*

Se podrían guardar como claves k, las imagenes y el polinomio ya que son necesarios para distribuir el secreto.

5. Según el documento se pueden guardar secretos de todo tipo (imágenes, pdf, ejecutables). ¿por qué?

Al realizar operaciones en GF(2^8) no se pierden datos por lo tanto se pueden guardar secretos en cualquier tipo de datos digital, ya sea imágenes, ejecutables, datos encriptados, señales de audio, video, entre otros.

Lo que se hace es encapsular los bytes correspondientes a estos archivos, en los píxeles de distintas imágenes, siguiendo el algoritmo.

La única diferencia sería que en vez de iterar por los píxeles de la imagen secreto, se debería iterar por los bytes del archivo a almacenar, incluido header, de manera que a la hora de obtener el secreto utilizando las sombras, se recuperen los bytes del binario en cuestión.

Sí es importante destacar que tendríamos que contemplar que el tamaño en bytes del binario sea divisible por el valor *k*, o establecer como implementación un padding al final del archivo por ejemplo.

6. ¿Cómo podría adaptarse la implementación realizada para poder guardar un archivo de imagen completo? (es decir guardando encabezado y pixeles)

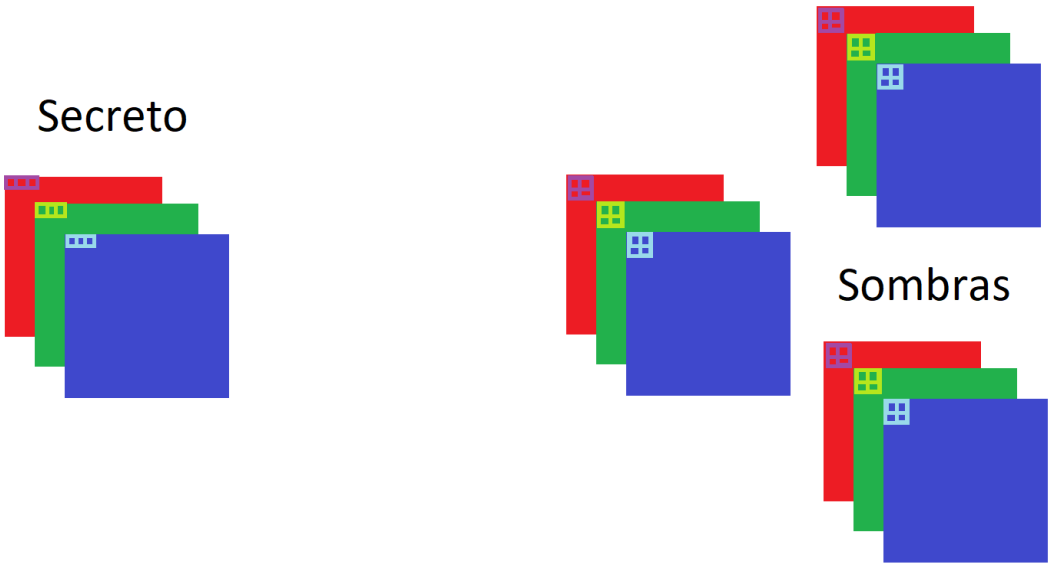
La implementación realizada sería prácticamente igual teniendo en cuenta las siguientes consideraciones:

- No discriminar el header de la imagen secreta.
- Las imágenes portadoras deben tener un tamaño mayor que la del secreto para que puedan albergar el header de la misma.
- A la hora de reconstruir la imagen, deberíamos reconocer dónde termina el header, lo cual en el caso de un archivo bmp se puede saber por el tamaño que ocupa.

7. Analizar cómo resultaría el algoritmo si se usaran imágenes en color (24bits por píxel)

Al utilizar imágenes a color se almacenaría la tabla de colores del header BMP y ahora al tener 24 bits por píxel la cantidad de bytes a almacenar como secreto será de 3 veces la cantidad de píxeles de la imagen.

Una posibilidad de implementación sería mantener el mismo algoritmo pero además iterando color a color RGB y se seguiría manteniendo la propiedad de que los XWVU se representarían en grupos de 4 pero esta vez según cada color.



El resultado serían sombras por color, donde contando con el canal R de todas las sombras, se recuperaría el canal R del secreto, y de igual manera con los otros 2 canales.

8. ¿Se podrían tomar los bloques de otra manera, en lugar de como matrices 2x2? Explicar.

X
W
V
U

Los bloques se podrían almacenar por ejemplo en matrices de 4x1, y mantener la misma proporción. Lo único que cambiaría es la distribución del secreto en la imágen, puesto que una de cada cuatro filas de la imágen sombra permanecerían inalteradas, suponiendo un esquema de la forma:

Lo importante es encontrar una configuración de matriz tal que la imagen pueda ser subdividida de esta manera, y decidir de qué manera distribuir los pares X, F(X) correspondientes a cada uno de estos bloques.

9. Discutir los siguientes aspectos relativos al algoritmo implementado: Facilidad de implementación y posibilidad de extender el algoritmo o modificarlo.

En cuanto a la implementación, la parte más compleja de desarrollar fue la de interpolación de polinomios en $GF(2^8)$. Se encontraron dificultades a la hora de manejar el "inverso multiplicativo" del cero ya que no tiene y eso ocasionaba la pérdida de algunos píxeles a la hora de recuperar la imagen.

El algoritmo se podría extender a imágenes a color ya que ahora únicamente se manejan imágenes en escala de grises.

10. ¿En qué situaciones aplicarían este tipo de algoritmos?

Este tipo de algoritmos se podría aplicar a situaciones en las que se requiere divulgar información confidencial entre varias entidades, para lo cual se requiere de un sistema que comparta dicha información de forma segura.

Por ejemplo, en una organización gubernamental o quizás no necesariamente en una empresa/organización. Si se posee una *Bitcoin wallet* de la cual miembros de una familia son propietarios y se quiere distribuir su clave, si cada uno de ellos tuviera una copia de ésta, sólo haría falta que una de las copias se viera comprometida para que se roben el contenido de la *wallet.* Por otro lado, si solo uno de los miembros conserva la clave, podría perderla o decidir traicionar a los demás.

En este caso, utilizando secreto compartido de Shamir, se crean tantas partes como miembros de la familia haya (supongamos n partes) y se establece un umbral para recuperar el secreto de n-1 partes, donde el secreto es la clave de la *Bitcoin wallet*.

De esta manera, la clave no está almacenada en un único lugar (lo que dificulta su robo), debe haber consenso entre los miembros de la familia para gastar los *Bitcoins* y, si un miembro pierde su parte, los otros pueden tranquilamente reconstruir la clave.