Criptografía y Seguridad TP
Grupo 5

1Q2023

Integrantes

- Baliarda Gonzalo 61490
- Pérez Ezequiel 61475
- Ye Li Valentín 61011

1. **Discutir los siguientes aspectos relativos al documento/paper.**

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

La organización del documento es adecuada, dado que inicia con una introducción al tema, luego trabajos pasados para dar un poco de contexto, y finalmente plantea el algoritmo novedoso.

También deja para el final demostraciones y conclusiones más específicas para los lectores interesados, pero no son fundamentales si sólo se quiere implementar el algoritmo.

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

La descripción de ambos algoritmos es bastante concisa y completa excepto en el último paso del algoritmo de reconstrucción de la imagen, donde dice “be the coefficients of x0 and x in fi(x) and gi(x) respectively”, sin aclarar el significado de x0 y x.

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

La notación es clara en general, pero hay que mirarla con detenimiento para seguir bien el uso de los diferentes subíndices que aparecen.

El único error que encontramos fue en la sección de resultados, donde pone erróneamente el cálculo para detectar sombras falsas (a+r*b en vez de r*a+b).

2. **El título del documento hace referencia a que es capaz de detectar sombras falsas(cheating detection). ¿cómo lo hace? ¿es un método eficaz?**

La construcción de las sombras se hace a través de dos polinomios, f y g. El método de detección de sombras falsas introduce un entero aleatorio que se usa para fabricar los primeros dos coeficientes del polinomio g, usando los dos primeros del polinomio f. Al recuperar el secreto, se chequea que este vínculo entre los polinomios mediante un entero aleatorio introducido siga existiendo.

El método es eficaz, dado que logra identificar hasta k-1 sombras falsas (lo que sería, que todos menos uno de los participantes necesarios mientan), con una probabilidad mayor al 99%.

3. **¿Qué desventajas ofrece trabajar con congruencias módulo 251?**

La desventaja que ofrece es que hay una limitación en la cantidad de tonos representables por cada pixel, pudiendo únicamente trabajar correctamente con tonalidades cuyo valor decimal se encuentre entre 0 y 250, dejando afuera el rango [251, 255] correspondiente a blancos muy puros.

Por esta razón, creemos que las imágenes que ocultamos y tienen muchos píxeles en blanco, al recuperarlas se ven con pixeles negros (los valores [251, 255] pasan a ser [0,5], es decir, tonalidades de negro muy oscuro).

Esto se puede observar en la imágen otorgada por la cátedra “Gandhishare”:

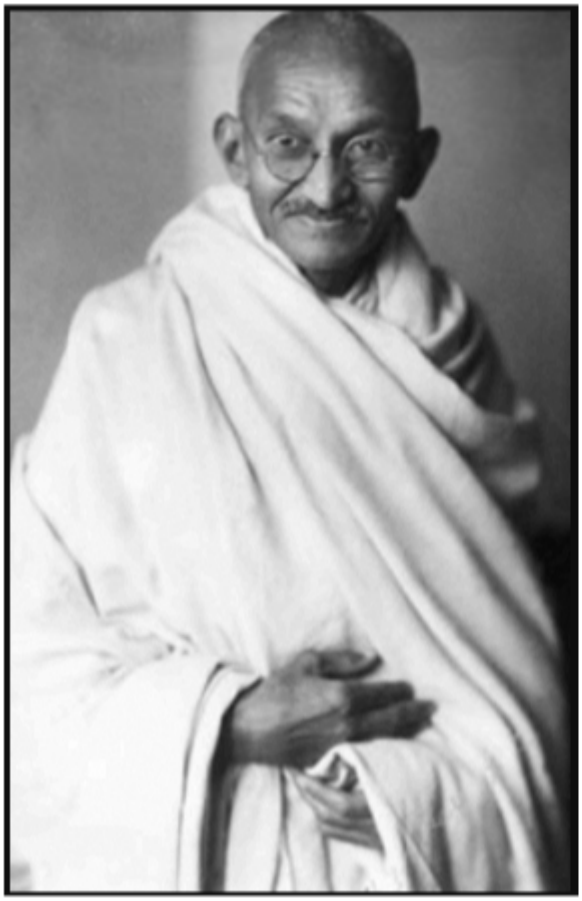

Figura 1.1 Imagen original

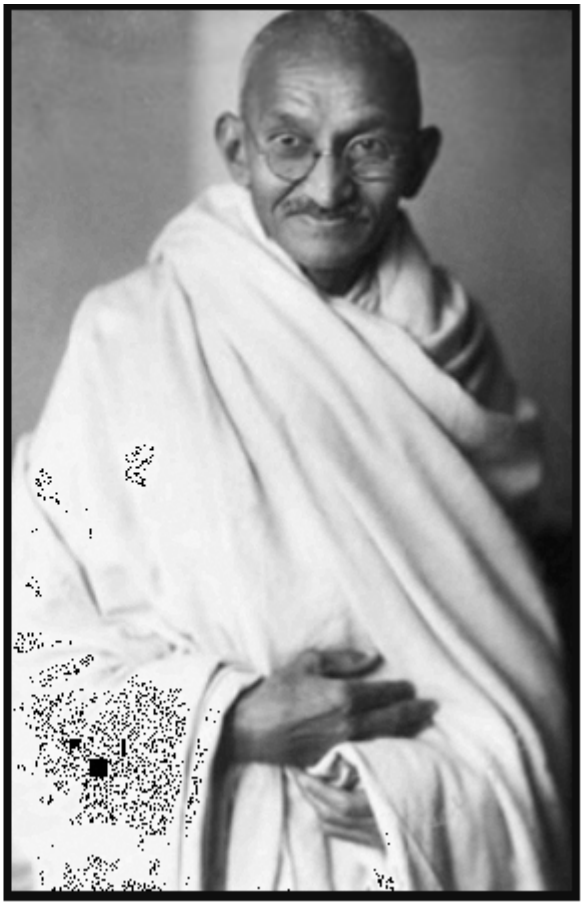

Figura 1.2 Imagen recuperada

4. **Con este método, ¿se podrían guardar secretos de cualquier tipo (imágenes, pdf, ejecutables). ¿por qué? (relacionarlo con la pregunta 3)**

Por defecto, con el algoritmo planteado no se podrían guardar secretos de cualquier tipo, ya que al trabajar con congruencia 251 se puede perder información de los bytes como se vió en la pregunta anterior.

En el caso de las imágenes, esto no es tan grave porque igual podemos interpretarlas, pero si se tratara por ejemplo de un ejecutable, muy probablemente no podríamos recuperarlo intacto.

Sin embargo, veremos en la pregunta 9 cómo se puede fácilmente atacar esta limitación para permitir guardar secretos de cualquier tipo.

5. **¿En qué otro lugar o de qué otra manera podría guardarse el número de sombra?**

Se podría guardar en cualquiera de los campos reservados del header BMP, o también dentro de la metadata (información adicional de los archivos BMP, que se encuentra entre el encabezado y los píxeles, la cual normalmente se usa para indicar la paleta de colores).

6. **¿Qué ocurriría si se permite que r, a0, a1, b0 o b1 sean 0?**

- a0 y a1 no pueden ser 0 porque se divide por ellos al hacer la detección de trampa al recuperar el secreto.
- Si r es 0, entonces b0 y b1 van a ser 0; pero si estos son 0, entonces siempre va a existir en la reconstrucción r=0 que cumple la siguiente condición; y perdemos la capacidad de detección de sombras falsas.

$$r = 0 \Rightarrow r * ai \ + bi = bi \equiv 0$$

- b0 y b1 no pueden ser 0, si ni a0 ni a1 ni r son 0; y ya vimos que estos últimos no pueden ser 0 en los ítems anteriores.

7. **Explicar la relación entre el método de esteganografía (LSB2 o LSB4), el valor k y el tamaño del secreto y las portadoras.**

El tamaño de las portadoras y del secreto será siempre el mismo (en cuanto a cantidad de píxeles). Sin embargo, lo que irá variando dependiendo del método de esteganografía (el cual a su vez dependerá del valor de k), es cuántos píxeles de las portadoras contienen realmente información del secreto.

Más en detalle, las relaciones directas son las siguientes:

- El método de esteganografía es LSB4 para k=3,4, y es LSB2 para k=5,6,7,8.
- La cantidad de píxeles usados de las portadoras se duplicará al usar LSB2 en vez de LSB4.

8. **Analizar cómo resultaría el algoritmo si se usaran imágenes en color (24bits por píxel) como portadoras.**

Por como está dado el algoritmo, si el tamaño de cada píxel es de 24 bits, entonces al calcular los coeficientes de los polinomios módulo 251 (donde cada

coeficiente es un pixel) se perdería toda la información de los primeros 2 bytes del pixel.

Si los píxeles se encuentren en formato RGB entonces la imagen que se recupere sería el mismo resultado que se obtiene con 8 bits pero en una gama de azules (último byte de los 3 usados para representar RGB) en lugar de grises.

9. Discutir los siguientes aspectos relativos al algoritmo implementado:

a. Facilidad de implementación

La dificultad de la implementación del algoritmo es media. Las principales dificultades que encontramos fueron las siguientes:

- Manejo preciso de los bits a la hora de embeberlos o recuperarlos de las portadoras.
- Implementación del método de Lagrange para interpolar puntos.
- Tener cuidado de realizar todas las operaciones módulo 251.

b. Posibilidad de extender el algoritmo o modificarlo.

Sería interesante poder usar este algoritmo para cualquier tipo de secreto (imágenes, PDFs, ejecutables, etc). Lo que se nos ocurrió para lograr esto, es implementar un paso previo al ocultamiento y posterior a la recuperación, consistente en un encoding/decoding del archivo respectivamente.

La idea sería encodear el secreto antes de ocultarlo, cosa de que el mismo no cuente con valores en el intervalo [251, 255] en sus bytes, que son los que se perderían. Una posible forma de hacer esto sería convertir el secreto en un stream de bytes donde cada byte use solamente los 7 bits más significativos. Así, 8 bytes del secreto original pasarían a ser 9 bytes después del encoding.
Ocultando el secreto encodeado,

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

Este tipo de algoritmos son muy útiles en situaciones en las que queramos dividir un secreto entre varias personas, con la propiedad de que varias de estas personas tengan que colaborar a la vez para poder recuperarlo sin poder de otro modo hacer nada con su parte individual.

Un posible ejemplo de aplicación, sería tener el código para lanzar un misil, y querer que se pongan de acuerdo al menos 4 de los 7 países que integran el G7 para poder lanzarlo. Así, se dividiría el código en un esquema (4,7), y se repartiría una portadora a cada país.

También sería conveniente si se quiere resguardar un secreto, pero sin ponerlo en un solo lugar centralizado. Podríamos en cambio repartirlo entre varios lugares, cosa de que si alguien encuentra una de las partes no le sirve para nada, y si perdemos el acceso a una de las partes tampoco pasa nada, dado que tendremos redundancia de sombras.