Trabajo práctico especial

72.44 - “Criptografía y Seguridad”

Secreto Compartido con esteganografía

2023 - 1Q

Grupo II

Integrantes:
Chao Florencia - Legajo 60054
De Luca Juan Manuel - Legajo 60103
Konfederak Sol - Legajo 60255
Cornidez, Milagros - 61432

21 de Junio de 2023

Cuestiones a analizar

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

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

La organización formal del documento parece adecuada. Comienza introduciendo el problema del cheating en los esquemas de secreto compartido y estableciendo cómo abordarlo. Luego presenta el esquema propuesto con detección de trampas y explica su funcionamiento. A lo largo del documento, se utilizan secciones y subsecciones claras para presentar y desarrollar diferentes aspectos del tema. Además, se citan referencias relevantes para respaldar los argumentos y proporcionar un contexto adecuado.

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

La descripción de los algoritmos propuestos en el paper son claros ya que comienza explicando los conceptos base para entender mejor el mecanismo y luego le suma complejidad al agregarle cheating detection. Es detallada ya que explica paso por paso lo que hay que ir haciendo, y es completa ya que explica como funciona y toma casos de ejemplo.

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

La notación utilizada es clara y se mantiene a lo largo del documento, sin embargo puede resultar un poco confuso de leer al utilizar muchos puntos (,...,) para expresar rangos de valores. También al momento de ver expresiones con muchos subíndices mezclados puede ser difícil de leer ya que no hace referencia a que estaría representando, como por ejemplo decir que *j* subíndice indica número de bloque. Nosotros no encontramos ningún error pero se puede destacar que utiliza ejemplos numéricos bastantes complejos de resolver a simple vista que escapa lo que está tratando de explicar.

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 detección de sombras falsas (cheating detection), se encuentra explicada en el paper. En el mismo, se comenta que el atacante puede obtener los polinomios $f^{**}(x) = f^{*}(x) + f(x)$ y $g^{**}(x) = g^{*}(x) + g(x)$ donde $f^{*}(x)$ y $g^{*}(x)$ son elegidos por el atacante, por lo que puede escogerse un número tal que $r^{*}a^{*}_{0} + b^{*}_{0} = 0$ y $r^{*}a^{*}_{1} + b^{*}_{1} = 0$. Por lo que, sí existe un r′ tal que

$r'(a_0 + a^*_0) + b_0 + b^*_0 = 0$ y $r'(a_1 + a^*_1) + b_1 + b^*_1 = 0$, no se puede detectar el cheating. Es decir, se puede detectar el cheating solamente cuando no exista un valor que satisfaga las ecuaciones $r_i * a^*_{i,0} + b^*_{i,0} = 0$ y $r_i * a^*_{i,1} + b^*_{i,1} = 0$.

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

La principal desventaja de trabajar con este módulo, es que no es resistente a ataques de fuerza bruta, ya que se puede recuperar con una probabilidad de éxito de $\frac{1}{251}$.

Pero, también se pueden encontrar otras desventajas, como la reducción del rango de valores. Es decir, el rango de valores se reduce de 0 a 250 y teniendo como consecuencia la limitación de representar números más grandes, obteniendo pérdida de información. También, genera una mayor repetición de valores en comparación con trabajar un espacio de números más grande. Sin olvidarnos, que al poder repetir información debido al pequeño espacio con el que nos encontramos trabajando, podemos generar patrones que revelen información y es algo indeseado.

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

Debido a que se trabaja con módulo 251, cualquier archivo es susceptible a perder información como consecuencia de esto. Al trabajar con archivos .bmp, en este caso, lo que se ve levemente modificado son algunos píxeles de la imagen. Pero, si nos encontramos trabajando con archivos donde la exactitud es relevante y cada bit es fundamental, no es recomendable utilizar este método. Mientras que, si se trabajan con archivos, donde teniendo una idea general del documento, se logra interpretar y obtener los resultados necesarios, se puede utilizar este método.

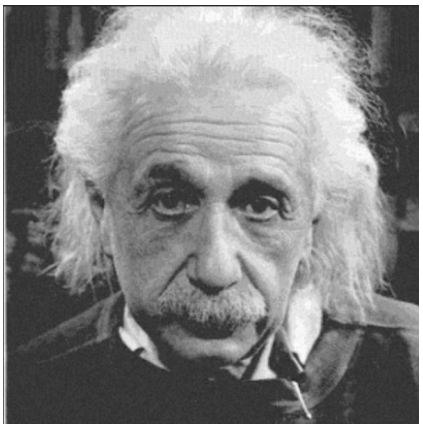

Imagen 1. Imagen distribuida

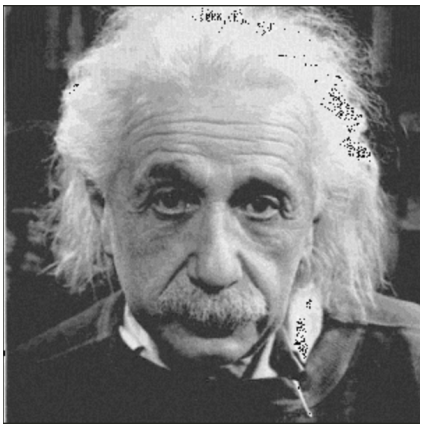

Imagen 2. Imagen recuperada

Se puede notar como al distribuir la *Imagen 1* y luego recuperarla hubo una pérdida de información que es considerada casi despreciable para la reconstrucción de la imagen original pero sí se quisiera ocultar archivos donde cada bit es relevante este método no sería el adecuado.

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

El número de sombra, podría guardarse en los primeros 4 bits de dos píxeles en específico, sin que se sobreescriba por LSB2 o LSB4. Hay un detalle a tener en cuenta, que es que, estos pixeles, se verán modificados respecto a la imagen original, ya que deben modificarse para almacenar el número de sombra.

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

Si r = 0, significa que la el polinomio para la compartición de secretos, es constante. Y como consecuencia, puede generar vulnerabilidades o debilidades en la protección del secreto compartido, ya que es una función determinística y se podría averiguar sencillamente el valor de la clave.

Si a_0 y a_1 son 0, el sistema de ecuaciones de cheating, no es invertible, por lo que no me permite calcular el valor de 2.

Y por último, si b_0 y b_1 son 0, implica que a_0, a_1 y r son 0; y por lo explicado anteriormente, no puede suceder.

Por lo que, que r, a_0, a_1, b_0 o $b_1$sean 0, compromete la seguridad y correcta funcionalidad del esquema.

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

Por consigna, sabemos que la relación entre k y el método de esteganografía es la siguiente; para k = 3 y 4, se utiliza LSB4 y para k = 5, 6, 7 y 8 se utiliza LSB2. En nuestro caso teníamos un conjunto de imágenes con un secreto escondido con k=3 y pudimos recuperar la siguiente imagen.

Imagen 3. secreto recuperado con k = 3

Luego probamos recuperar el secreto con k=4 y obtuvimos el siguiente resultado

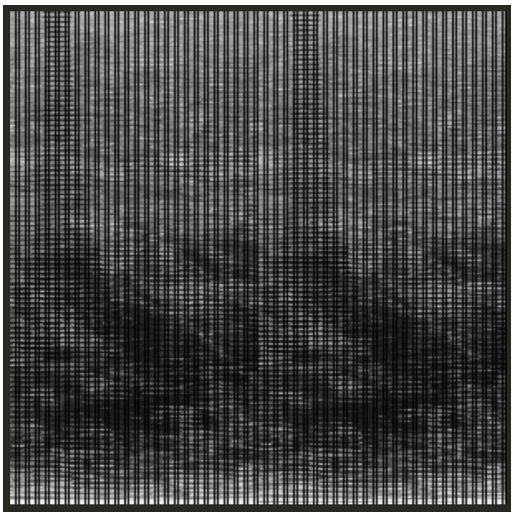

Imagen 4. archivo de salida con k = 4

Con algún otro *k* el programa devuelve error ya que corresponde otro LSB.

En el caso de LSB4, con k = 3 o 4, se utilizaron 3 o 4 pixeles consecutivos para ocultar un bit del secreto. Por lo que, se reemplazaron los 4 bits menos significativos de cada píxel con los obtenidos anteriormente.

Mientras que, en el caso de LSB2 con valores de k = 5, 6, 7 u 8, se utilizaron 5, 6, 7 u 8 pixeles consecutivos para ocultar un bit del secreto. Por lo que, se reemplazaron los 2 bits menos significativos de cada píxel con los obtenidos anteriormente.

Al utilizar LSB4 con k=3 o 4, se requiere menos pixeles para ocultar la misma cantidad de información que con LSB2. Por lo que, puede resultar útil, si la información a ocultar no es tan sensible o se necesita mayor rapidez. Además, utilizar LSB4 puede resultar útil cuando el tamaño de las portadoras se encuentra limitado. Utilizar LSB2 con k = 5, 6, 7 u 8, nos permite obtener una mayor seguridad respecto a la ocultación del secreto ya que minimiza el impacto visual de la transformación y así, poder evitar la detección del secreto oculto. Mientras que, para utilizar LSB2, se tiene que tener a disposición un mayor tamaño de portadoras.

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

Si bien el algoritmo de secreto compartido de Shamir es independiente del tipo de dato que se esté utilizando como portador, hay algunas consideraciones que se deberían tener en cuenta si se quisiera que esta implementación funcionara para imágenes a color. En primer lugar, el tamaño del secreto será mayor. Por otro lado, se debe tener en cuenta a la hora de recuperar el secreto, que cada píxel de una imagen a color consta de tres componentes de color (rojo, azul y verde), por lo que deberan hacerse por separado las operaciones de recuperacion del color. Por último, es posible que la carga computacional a la hora de realizar operaciones sea mayor.

9. Discutir los siguientes aspectos relativos al algoritmo implementado:

Facilidad de implementación

La implementación del algoritmo no fue compleja en sí. La complejidad estuvo en comprender el algoritmo propuesto en el paper, entendiendo bien la detección del cheating y su solución al respecto. Luego, al utilizar C, mucho tiempo fue destinado al manejo de memoria y de punteros que tal vez en otros lenguajes de programación no es necesario.

Posibilidad de extender el algoritmo o modificarlo

Se podría extender/modificar el algoritmo para que acepte otros tipos de archivos, o para que acepte imágenes .bmp pero con diferente cantidad de bits por píxel (imágenes a color). También se podría mejorar para que utilice valores de k mayor a 8 para ocultar mayor cantidad de pixeles y así esconder mejor la imagen.

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

El algoritmo es útil cuando se requiere que un secreto no sea conocido solo por una persona o entidad. Algunos casos de uso comunes pueden ser los siguientes.

- Almacenamiento seguro de contraseñas.

- Protección de archivos confidenciales.
- Acceso a datos sensibles en organizaciones.