INFORME TPE CRIPTOGRAFÍA Y SEGURIDAD 2015

GRUPO 5

Integrantes:

- Mehdi Tomas, 51014
- Nicolas Buchhalter, 53008
- Leonel Badi, 51297
- Agustina Fainguersch, 50589

Discutir los siguientes aspectos relativos al documento de Thien y Lin:

Introduccion

El paper comienza presentando el problema de tener secretos en un solo archivo y propone como solución, la alternativa de secreto compartido. A su vez, dice que se utilizará una variación derivada del esquema de umbral (r,n)

Revisión del esquema de umbral (n,k)

En esta sección se describe brevemente cómo funciona el esquema de umbral (r,n)

El método de compartir imagenes secretas

En este subtítulo se habla de tres temas tratados como etapas: la etapa de compartir, la etapa de revelar y dos maneras de compartir sin pérdida de información.

Resultados del experimento

Se presenta el resultado del algoritmo utilizado sobre imagenes conocidas.

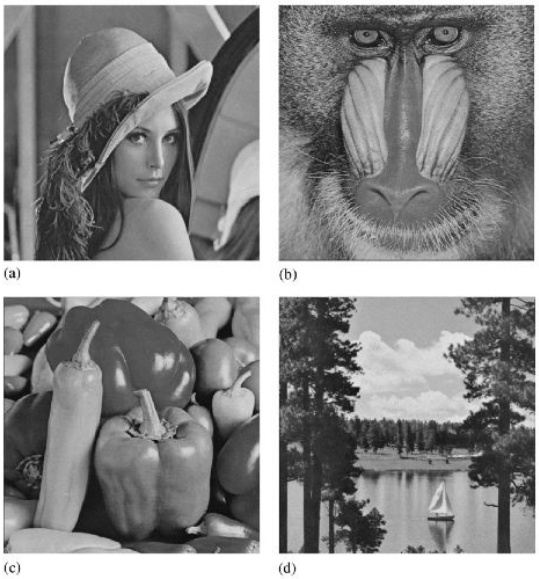

Fig. 2. Hiding shadows in some other existing images: (a)–(d) the images obtained after embedding the shadows Fig. 1(c)–(f), respectively, in the images: Lena, Monkey, Peppers, and Scene.

(imagen tomada del paper Computers & Graphics 26 (2002) 765–770)

Análisis de seguridad

Se realiza la demostración matemática de que con r-1 sombras la probabilidad de conseguir la imagen secreta es de (1/251)^(imageWidth*imageHeight/r)

Los beneficios de la reducción de tamaño de la propiedad de las imagenes sombra

En esta parte del paper se comenta los beneficio de que las sombras de la imagen secreto sean más pequeñas que la imagen secreta tiene varias ventajas. Por ejemplo para almacenarla de manera distribuida. También se habla de la ventaja de este método para que los datos no sean destruidos o perdidos.

Conclusiones

El paper presenta un método para distribuir una imagen secreta en varias sombras de tamaño 1/r. También presenta 2 manera de solucionar la perdida de informacion para los casos que hay que truncar valores mayores a 250. Y comenta la posibilidad de almacenar imagenes en varios repositorios gratuitos de manera distribuida y las ventajas de esto.

Algoritmo de distribución:

1. Truncate all gray values larger than 250 to 250 so that the gray values of the secret image are in the range 0–250.
2. Use a key to generate a permutation sequence to permute the pixels of the secret image.
3. Sequentially take r not-shared-yet pixels of the permuted image to form a section.
4. Use the section in Step 3 and Eqs. (3) and (4) to generate n pixels for the n shadow images.
5. Repeat Steps 3 and 4 until all pixels of the permuted image are processed.

$$q_j(x) = (a_0 + a_1 x + \cdots + a_{r-1} x_{r-1}) \bmod 251, \qquad (3)$$

$$q_j(1), q_j(2), \ldots, q_j(n). \qquad (4)$$

Algoritmo de recuperación:

1. Take the first non-used pixel from each of the r shadow images.
2. Use these r pixels (a subset of $\{q_j(1), \ldots, q_j(n)\}$) and the Lagrange's interpolation to solve for the coefficients $a_0 - a_{r-1}$ in Eq. (3). The coefficients $a_0 - a_{r-1}$ are then the corresponding r pixel values of the permuted image.
3. Repeat Steps 1 and 2 until all pixels of the r shadow images are processed.
4. Applying the inverse-permutation operation to the permuted image to get the secret image.

La notación a lo largo del documento es consistente, pero es diferente a la utilizada en clase. Reemplaza los k's por r's.

¿Cómo se puede evitar el truncamiento de píxeles mayores que 250 según el documento? ¿Podría haber otra alternativa?

Opción 1: La idea es que en los casos que es menor o igual que 250 se almacena ese número y en caso de que sea mayor que 250 se almacenan dos números. 250 y el número - 250 de manera contigua. Luego se crean secciones de 2 números.

Opción 2: Utilizar imagenes que tengan todos los pixeles menores a 250, de esta manera el truncamiento no se utilizará nunca.

Explicar el criterio utilizado para elegir imágenes portadoras en el caso de k distinto de 8. Indicar si hubo propuestas previas que fueron descartadas.

A la hora de elegir las imagenes portadores para k distinto a 8, no tomamos ningún criterio dado que no nos hace ninguna diferencia por la manera que se oculta la información en las sombras.

Explicar el criterio utilizado para ocultar las sombras en el caso de k distinto de 8. Indicar si hubo propuestas previas que fueron descartadas.

En el caso de k distinto de 8, lo que se decidió fue agarrar las mismas imágenes portadoras pero en vez de modificarles el bit menos significativo de cada byte, se modificaron los 8/k bits menos significativos. Por esto es que solo permitimos usar valores de k que sean divisores de 8 (2, 4, 8). De esta forma a medida que se usan k más pequeños, las sombras se ven más modificadas con respecto a las originales, pero como son los bits menos significativos, las imagenes siguen siendo nítidas. Si bien este fue el método seleccionado en un principio habíamos pensado hacerlo de otra manera.

La otra manera era poner 1 bit cada K/8 bytes, es decir que si K era menor que 8, por ejemplo 4, se pone 1 bit cada medio byte. De esta manera podria seria mas facil generalizar y poder utilizar cualquier K(inclusive mayor que 8). El problema con esta propuesta era que para valores de K menores que 8, la perturbación en las sombras se hacía muy evidente dado que no solo tocaba los bits menos significativos.

Con el método implementado pudimos modificar las sombras teniendo una perturbación casi imperceptible al ojo humano.

Discutir los siguientes aspectos relativos al algoritmo implementado:

Facilidad de implementación

La facilidad de implementación del algoritmo recae principalmente en la facilidad con la que se puedan manipular los bytes y bits en el lenguaje a utilizar. En C es algo bastante simple el manejo de bits por lo que se vuelve bastante sencillo. Luego está la dificultad de resolver los sistemas de ecuaciones, que al plantearlo a modo de matriz y diagonalizarla se volvió un algoritmo bastante sencillo.

Un buen punto a destacar es lo fácil que se hace el algoritmo de implementar cuando lo que se está buscando es poder ocultar una imagen en otras sin que siquiera se pueda notar en el ojo humano.

Posibilidad de extender el algoritmo para que se usen imágenes en color.

Las imágenes de color son iguales a las blanco y negro solo que por cada pixel, tienen 3 bytes, uno que representa el color rojo, otro el color verde y otro el color azul. De esta

forma se podria hacer lo mismo , esconder cada bit de cada byte en un byte diferente de otras imagenes de color, con la correspondencia de que cada byte rojo se distribuya en los bytes rojos de las imagenes y asi con el resto de los colores. De esta forma uno escondería en vez de un bit de cada byte, un bit de cada 3 bytes.

De ser implementado de esa manera(el rgb en las imagenes), el algoritmo seria el mismo. De ser otra la implementación de los 3 canales habría que modificar el algoritmo de manera que por cada pixel haga el algoritmo ya implementado accediendo a las 3 distintas capas de color.

¿Qué dificultades tuvieron en la lectura del documento y /o en la implementación?

En la lectura del documento no se presento ninguna dificultad dado que la catedra habia explicado con anterioridad el tema.

Las dificultades surgieron al momento de la implementación y especialmente con las funciones de muy bajo nivel. Estas funciones son las que tenían que utilizar máscaras para realizar una correcta modificación de los bits. Si bien a simple vista esta es una tarea simple, la dificultad residio en el hecho de que lo hicimos genérico para k=2,4,8, utilizando solo una función y haciendo uso del operador de “shifteo” para ir modificando la máscara de acuerdo a los parámetros de entrada.

Otra dificultad fue el hecho de no tener una imagen desencriptada para poder compararla con la salida del algoritmo, es decir no poder tener la salida esperada.

¿Qué extensiones o modificaciones harían a la implementación o al algoritmo?

A este algoritmo se le podria agregar la capacidad de poder realizar distribuciones y recuperaciones con un valor de K mayor que 8. Esto se haría modificando un bit de cada K/8 bytes.

Ademas se le podria agregar una passphrase asociada a cada imagen para evitar que en el caso de hurto el secreto pueda ser recuperado.

Otra mejora sería agregar un canal de comunicación utilizando el espacio que hay entre el header y el comienzo de la imagen. De esta manera ahi se podria colocar información que luego sea necesaria para descifrar la imagen.

¿En qué situaciones aplicarían este tipo de algoritmos?

Hay diferentes escenarios que se pueden plantear al momento de recurrir a estos algoritmos:

El primero es cuando no se puede asegurar el resguardo de una única clave, es decir que esta clave puede ser hurtada o perdida. Si esto sucediera o bien la información estaría comprometida o pérdida. Distribuyendo N claves compartidas, podemos bajar la probabilidad de que esto suceda notablemente.

El segundo escenario es cuando es imprescindible que una sola persona no tenga acceso a ninguna parte de la información. Cuando nos referimos a ninguna parte es que la información tenga secreto perfecto a no ser que se posea K claves. En este caso, ningún portador de la clave va a poder acceder a la información por su cuenta, sino que siempre va a tener que requerir la colaboración de otros K-1 individuos.