INFORME

Trabajo Práctico

Secreto Compartido en Imágenes con Esteganografía

72.44 - Criptografía y Seguridad

1er cuatrimestre 2019

- **Integrantes:** Lóránt Mikolás**,** Diego Bruno Cilla y Agustín Calatayud.
- **Número de grupo**: 5

- **Fecha de vencimiento de la entrega:** 24/06/2019

1. Índice

2. Introducción

El objetivo principal de este trabajo práctico es implementar un programa que permita la distribución y recuperación de un secreto compartido en imágenes con esteganografía. Los algoritmos y metodologías utilizadas están basadas en el documento escrito por Kiki Ariyanti y Nurfathiya Faradiena Azzahra de Universitas Indonesia llamado “Verifiable Image Secret Sharing Using Matrix Projection”. Este a su vez es una mejora del trabajo presentado por Li Bai y Saroj Biswas, de la Universidad Temple de Filadelfia titulado “An Image Secret Sharing Method”.

Entonces las dos principales funciones del programa son: la distribución de un archivo secreto (una imagen) de extensión” .bmp” en otras que funcionarán como sombras del esquema (k,n) y la recuperación de ese secreto en un nuevo archivo “.bmp” nuevo a partir de por lo menos k sombras.

Vale la pena mencionar que se trata de un programa implementado en el lenguaje de programación “C” y su ejecución se realiza por la terminal de comandos (para detalles sobre la la modalidad de ejecución referirse al README o al anexo).

En este informe se explicarán los argumentos detrás de los algoritmos, se hará un análisis profundo de los mismos y de sus alternativas, se explicitan las decisiones tomadas a lo largo del desarrollo, dificultades encontradas y aplicaciones posibles del esquema.

3. Fundamentos

A continuación se explicarán los dos algoritmos utilizados y algunos detalles acerca de la notación. Se recomienda leer ambos documentos previo a la lectura del presente informe.

En la base del esquema (k, n) se encuentra el hecho de que el secreto distribuído en n partes únicamente puede ser recuperado y determinado si se conoce por lo menos k de ellas.

Se deben tener en cuenta dos aspectos: si n es muy pequeño se corre el riesgo de que al perder alguna parte se haga imposible recuperar el secreto y si n es muy grande corremos mayor riesgo de que alguna parte sea filtrada. Para ocultar las partes se utilizará esteganografía de manera que un adversario no pueda encontrarlas fácilmente. En este caso se esconderán las diferentes partes del secreto en imágenes “.bmp”.

Vale la pena notar que en esta implementación también se cuenta con una marca de agua que sirve para detallar si la imagen fue recuperada correctamente o no.

En las siguientes secciones se detallarán el algoritmo de distribución y recuperación basados en la proyección de matrices de Li Bai.

3.1. Algoritmo de distribución

Para distribuir, se deben crear las partes de la imágen secreta. Se aclara que todas las operaciones detalladas se realizan módulo 251. Un detalle a notar es que al leer los valores de la imagen a ocultar se decidió tomar los valores por encima de 250 como 250 para no dar vuelta los valores con las operaciones modulares.

En primer lugar, se recibirá el input del usuario con respecto a los valores de k y n que se desean utilizar. Sin embargo, se deben hacer ciertas validaciones: k no puede ser mayor que n, ambos deben ser número enteros positivos y $k \varepsilon [2, 251]$. Luego se leen las matrices S de la imágen (imágenes de n x n). A partir de ahora la explicación del algoritmo será aplicado a cada matriz S, es decir que se realizará con cada matriz de n x n.

En segundo lugar, se genera una matriz aleatoria A de tamaño mxk y en particular se tomó m = n debido a que se tiene que cumplir la restricción m> 2 (k – 1) – 1.

En tercer lugar, se calculó tal como lo indica el paper la matriz Sdoble y R en Z_{251}.

En cuarto lugar, se generaron las n matrices Xj de k x 1 . Estas matrices son linealmente independientes. Para esto se creó un array de valores '*a'* random (no repetidos) en Z_{251} y cada uno de Xj se le asignó un 'a' que fue elevado a un exponente manera creciente (0,1,2 … k-1) para llenar las componentes del *array*.

En quinto lugar se calcularon los vectores Vj como producto de A y Xj. No se entrará en mayor detalle ya que el procedimiento imita aquel que se expone en el documento.

Luego, se computaron las matrices Gj, la parte más difícil del algoritmo, cada columna de esta matriz resulta de multiplicar una columna de R con el número de sombra que corresponde elevado a un exponente.

Una vez computadas las matrices G se computan las matrices Sh, estas son la unión de las matrices V y G de la siguiente forma Shj = [Vj Gj].

Por último, se selecciona la imagen indicada como marca de agua y se computa Rw tal como lo indica el documento: Rw=(W - $\mathbb{S}$) (mod p) y se hace pública guardando la misma en un archivo de extensión ".bmp" para su acceso en el momento de la recuperación.

En este momento se comienza la etapa de esteganografía donde se toman todas las matrices Shj para todas las matrices S y se juntan según su índice. Luego estas matrices se

reparten en las distintas imágenes, según su esquema (k, n) se realiza el procedimiento de manera diferente. Si se trata del esquema (2, 4) se repartirán dos bits de datos de la matriz en los dos bits menos significativos de cada byte en la imagen donde se la quiere repartir. El esquema (4, 8) en cambio repartirá un bit en los el bit menos significativo de cada byte en la imagen donde se la quiere repartir.

3.2. Algoritmo de recuperación

Para recuperar el secreto original se deben elegir las k imágenes que serán usadas para hacerlo. Para ello primero se separan las matrices Sh extraídas de las imágenes en las diferentes partes para facilitar su manipulación.

En primer lugar, se computa B tal como lo indica el documento. B está compuesto por los k vj ubicados de la siguiente forma: B = [v1 … vk] ordenados según el número de sombra que sean.

En segundo lugar, se construye la matriz Sdoble realizando la proyección de B módulo 251.

En tercer lugar, se genera la matriz G obteniendo la segunda parte de cada Sh y con esas matrices se recupera R. Esto se hace mediante la resolución del sistema de ecuaciones siendo la matriz de coeficientes la compuesta por el número de sombra elevado a un exponente y la matriz de resultados la matriz G correspondiente.

En cuarto lugar, se vuelve a conseguir el secreto haciendo la suma de Sdoble y R.

Para finalizar se crea la marca de agua W para verificar que el proceso se realizó correctamente haciendo la suma de Rw (pública) y Sdoble.

3.3. Análisis del documento

El documento en general resulta complejo de seguir, tanto como está escrito como también las explicaciones de los algoritmos. En múltiples ubicaciones no se explica bien qué sucede cuando se realiza el algoritmo con distintos n y k. La notación en el documento de Ariyanti y Azzahra es confusa, con errores e inconsistencias. Cuando realiza las matrices G habla de I cuando debería estar hablando de R por ejemplo. Además cuando se explica la recuperación no se explica la forma en la que se recuperan las matrices G de forma explícita.

En cuanto a la mejora que plantea el algoritmo de Ariyanti y Azzahra frente al de Li Bai, esta es la posibilidad de verificar si el secreto extraído mediante el algoritmo es el correcto. Si se verifica la marca de agua original y la que resulta del algoritmo se puede

verificar que el secreto que retorna el procedimiento es el correcto. Esto podría ser fundamental no solo para verificar el correcto funcionamiento sino que se tiene el secreto correcto y no es un intento de sabotaje.

4. Análisis de los algoritmos

Es de interés análisis algunos aspectos específicos de los algoritmos como puede ser la aplicación de ciertas restricción o la toma de decisiones. Aquellas serán presentados en esta sección.

4.1. Dificultades al elegir pares (k,n) distintos

Las dificultades para elegir los pares (k,n) surgen por dos factores, uno es el hecho de que la cantidad de bytes de la imagen del secreto y la marca de agua deben ser divisibles por n * n. Otro factor que dificulta la elección de los pares es la esteganografía, se deben tener en cuenta los pares elegidos ya que afecta la cantidad de bits en los que se debe distribuir las partes (por cada byte de las imágenes de distribucion).

4.2. Rango de A y de A^t . A

Resulta importante controlar el rango de A ya que si no fuera de rango k el procedimiento que calcula la proyección será diferente. De esta forma se garantiza que $proj(A) = A * (A^tA)^{-1} * A^t$. Para garantizar esto también es importante controlar el rango de A^tA porque se requiere que la misma sea invertible y por lo tanto de rango completo.

4.3. Válida forma de generar los *Xi*

La forma de generar los distintos Xi genera vectores linealmente independientes porque al elegir un número aleatorio y teniendo como primer componente del vector igual en todos los Xi se puede verificar que son LI. Cómo a partir de la segunda componente de Xi son números distintos, claramente la única forma de que sumen 0 es que el coeficiente que multiplica a cada vector sea 0 (es decir son linealmente independientes).

4.4. Imagen *Rw*

Con respecto a la imagen Rw, la misma puede ser ocultada mediante esteganografía pero no es necesario ya que no se puede extraer de ella ninguna información del secreto. En el caso de querer hacerlo se podría distribuir la imagen Rw en una imagen con 8 o 4 veces más bytes que la original (si se quiere distribuir en un solo bit o en dos respectivamente).

4.5. Detalle del n

Vale la pena notar que siempre se debe saber el n con el que se está trabajando, incluso cuando se recupera, porque al dividir los bytes repartidos se debe saber el tamaño de las matrices a utilizar. Esto es debido al funcionamiento específico del algoritmo de distribución y recuperación.

4.6. Alternativas de guardado de número de sombra

El número de sombra se podría guardar en cualquier parte del bmp fuera del header y los bytes que representan los pixeles. Se debe tener en cuenta modificar el offset de donde empiezan los pixeles si se quiere incluir en el medio.

5. Implementación

En esta sección se abordarán las características del proceso de desarrollo incluyendo algunas de las dificultades encontradas y la posibilidad de expandir la implementación en el futuro.

5.1. Facilidad de implementación

Con respecto a la implementación, se la considera sencilla en relación a la complejidad del algoritmo. Es decir, una vez comprendido el algoritmo su implementación no fue complicada. Como se indicó anteriormente los documentos en los cuales se detalla su funcionamiento no son claros. Además al poder realizar la implementación de cada uno

de los módulos necesitados para la ejecución por separado fue posible probarlos y testearlos independientemente previo a la integración.

En cuanto a la integración se presentaron algunos errores por diferencias en la parametrización pero una vez resueltos no surgieron mayores problemas.

Un factor que se tuvo que tener en cuenta en todas las partes de la implementación es el manejo de memoria. Por ejemplo cada vez que se realiza una operación entre dos matriz, el resultado es guardado en una nueva matriz y es importante asegurarse de liberar la memoria que se deja de utilizar. Además el programa es compilado con *flags* que permiten por ejemplo detectar variables inutilizadas o convertir detalles que de otra manera serían *warnings* (avisos) en errores de compilación. Por último, uno de los *flags* detalla la versión del compilador de gcc que debe ser utilizada de manera que todos los módulos sean consistentes entre sí y asegurar que el código escrito pueda ser ejecutado en *pampero*.

5.2. Mantenibilidad del algoritmo

En cuanto a la mantenibilidad del algoritmo no debería ser de gran dificultad expandirlo o modificarlo para que alcance otros casos ya que su implementación se encuentra desacoplada y modularizada. Para empezar se cuenta con una librería de operaciones modulares sobre matrices (independiente del resto del programa) que permite realizar sobre ellas todas las acciones necesarios. Por otro lado contamos con una librería para la generación de números random enteros. También se tienen librerías independientes y desacopladas para la realización de la esteganografía y el IO sobre archivos “.bmp”. Por último, se encuentran los dos pilares del programa que son los algoritmos de distribución y recuperación que hacen uso de las partes mencionadas anteriormente. Además se cuenta con archivo que lee las opciones elegidas en la ejecución por la terminal y determina cuál de las dos acciones principales se desea realizar y con qué parámetros (que son a su vez validados).

5.3. Aplicaciones

Este tipo de programa tiene aplicaciones por ejemplo si se quiere mantener un secreto disponible online pero no se quiere que si el servidor donde se almacena es vulnerado, que se pueda extraer el secreto. Entonces se puede distribuir el secreto en múltiples servidores con este método y aunque algún servidor deje de funcionar se podrá recuperar el secreto mientras que si uno es vulnerado no se puede recuperar la información.

Cualquier tipo de esquema donde se necesiten múltiples personas para acceder a información confidencial pueden ser uso de este tipo de algoritmos.

6. Conclusiones

En conclusión, fue posible implementar un programa que sirva para la distribución y recuperación verificable de un secreto compartido en imágenes con esteganografía. basados en los documentos de Kiki Ariyanti, Nurfathiya Faradiena Azzahra y Li Bai.

Esta implementación resultó eficiente y modular aunque se hayan tenido problemas con la comprensión del algoritmo detallado en el documento. Resulta una aplicacion muy util si se quiere ocultar un secreto, sobre todo si se requiere un número mínimo de personas para verlo.

7. Anexo

7.1. Manual de ejecución

Los pasos para ejecutar el program son los siguientes:

1. Abrir una terminal de comandos.
2. Desplazarse con la ayuda del comando cd al directorio raíz del proyecto.
3. Ejecutar el comando "*make all*" esto generará un ejecutable en el directorio actual denominado "ss".
    a. Si posteriormente se desea eliminar el ejecutable y los archivos objeto se puede ingresar: "*make clean*".
4. Ingresar el comando "*./ss -h*" para obtener los parámetros que pueden ser utilizados en el programa y la forma en que deben ser ingresados.
    a. Un ejemplo de ejecución sería:
    "./ss –d –s Albert.bmp –m Paris.bmp –k 4 – n 8 –i color280x440/ " (ejemplo basado en el planteado por la cátedra)
    b. Con respecto a los parámetros sugeridos por la cátedra se cambió el nombre del parámetro "-dir" por "-i". Se referencia del enunciado proveído por la cátedra la lista de parámetros posibles con la modificación:

- -d o bien –r
- -s imagenSecreta
- -m imagenMarca
- -k número
- -n número
- -i directorio
- -v
- -h

Significado de cada uno de los parámetros obligatorios:

- -d: indica que se va a distribuir una imagen secreta en otras imágenes.
- -r: indica que se va a recuperar una imagen secreta a partir de otras imágenes.
- -s imagen: El nombre imagen corresponde al nombre de un archivo de extensión .bmp. En el caso de que se haya elegido la opción (-d) este archivo debe existir ya que es la imagen a ocultar y debe ser una imagen en blanco y negro (8 bits por pixel) Si se eligió la opción (-r) éste archivo será el archivo de salida, con la imagen secreta revelada al finalizar el programa.
- -m imagen: El nombre imagen corresponde al nombre de un archivo con extensión .bmp. En el caso de que se haya elegido la opción (-d) este archivo es una imagen en blanco y negro que servirá como “marca de agua” para verificar todo el proceso. Debe ser de igual tamaño que la imagen secreta. En el caso de que se haya elegido la opción (-r) este archivo es una imagen en blanco y negro que contiene la transformación de la imagen de “marca de agua”.
- -k número: El número corresponde a la cantidad mínima de sombras necesarias para recuperar el secreto en un esquema (k, n).
- -n número: El número corresponde a la cantidad total de sombras en las que se distribuirá el secreto en un esquema (k, n).
- -i directorio: El directorio donde se encuentran las imágenes en las que se distribuirá el secreto (en el caso de que se haya elegido la opción (-d)), o donde están las imágenes que contienen oculto el secreto (en el caso de que se haya elegido la opción (-r)). Debe contener imágenes de extensión .bmp, de 24 bits por pixel. “

Significado de los parámetros no obligatorios:

- -v: indica modo verbose.
- -h: indica ayuda. Imprime un mensaje con los comandos disponibles y cómo ingresarlos.

7.2. Comentarios adicionales

Cabe destacar que al realizar la recuperación del secreto con las imágenes de prueba encontramos unos parches de pixeles en negro donde probablemente había blancos en el secreto original. Una explicación que encontramos de esto es que el algoritmo de distribución utilizado para las mismas no tiene el mismo recaudo que el nuestro y al encontrar valores por encima de 250 no los trata de forma diferente por lo que al realizar operaciones modulares, valores muy blancos se vuelven negros. Y es por esto que al recuperar el secreto conseguimos imágenes con parches negros.

8. Bibliografía

La bibliografía utilizada fue aquella que brindó la cátedra:

1. N F Azzahra and K A Sugeng 2018 J. Phys.: Conf. Ser. 1108 012082
2. *An Image Secret Sharing Method* by Li Bai, Saroj Biswas, ECE Department Temple University Philadelphia, PA, U.S.A.