UNIVERSIDAD
NACIONAL
DE COLOMBIA
Revista de la Facultad de Medicina 2000; 49 (1):295-301

¿Puede ocultarse información en una imagen de computador?

Jorge Eduardo Ortiz Triviño, Profesor Asistente, Departamento de Ingeniería de Sistemas, Universidad Nacional de Colombia.
E-mail: jorgeo@ing.unal.edu.co

INTRODUCCIÓN

Con la actual tecnología computacional es posible implementar técnicas de procesamiento de señales que permitan ocultar información dentro de archivos existentes. De esta manera es posible, por ejemplo, emplear un archivo gráfico para guardar un texto dentro de él sin que la imagen contenida en éste se dañe o se deteriore y con la ventaja de que el texto oculto puede recuperarse. Este tipo de procedimientos es útil en seguridad de redes y permitirá realizar transacciones seguras en la red Internet, entre otras aplicaciones.

En este artículo se propone una técnica y se explica el programa prototipo que logra realizar esta tarea (programa implementado por estudiantes de la carrera de Ingeniería de Sistemas).

Esteganografía

La palabra esteganografía viene del griego *stegos*, que significa cubierta, por lo que esteganografía significa escritura oculta. De esta manera, la esteganografía puede entenderse como el conjunto de técnicas que permiten ocultar o camuflar cualquier tipo de datos dentro de otros datos. Por ejemplo, es posible incluir en una imagen de computador el texto de un cuento de Rafael Pombo sin que el usuario desprevenido detecte que la imagen que está observando contiene esa información adicional que podrá extraer si conoce el método.

Es conveniente aclarar la diferencia entre criptografía y esteganografía estricta. La criptografía modifica los datos para que no sean legibles, a diferencia de la esteganografía, que simplemente los toma y los oculta entre otros datos. Sin embargo, hoy en día, con los computadores, ambas técnicas son combinables, logrando asi una seguridad mayor. Es de esta última manera como realmente es fuerte la esteganografía. El entorno más favorable para la esteganografía es aquel en el que se realicen búsquedas automatizadas de los mensajes, o en los que no sea una técnica demasiado conocida con el apoyo, naturalmente, del computador.

Algo de historia

La esteganografía tiene un origen antiguo, técnica que, como es de suponerse, nació como una herramienta para espionaje (hasta se ha llegado a especular que la misma Biblia está plagada de estas técnicas y, cuenta también la historia, que fue el mismo Newton quien, basado en procedimientos matemáticos, intentó extraer códigos e información oculta en este libro). A lo largo de la Historia los Estados fueron desarrollando técnicas que les permitieran comunicarse con sus espías en el extranjero sin que las autoridades del país que acogía al agente pudieran descubrir que estaba teniendo lugar esa comunicación. Caso similar ocurrió con las organizaciones secretas o clandestinas, que las utilizaron para comunicarse sin ser descubiertos y sin poner en peligro a sus miembros.

Los escritores judíos tambien solian usar estas técnicas ocultando el significado de sus textos invirtiendo el alfabeto, es decir, utilizaban la última letra de éste en lugar de la primera, la penúltima en vez de la segunda, y así sucesivamente. Este sistema, tambien aparece en la Biblia. Los éforos de Esparta se comunicaban con sus generales de campo por medio de mensajes escritos en los extremos de una banda de cuero que se enrollaba formando una espiral sobre un bastón. Una vez desenrollado, el mensaje sólo podía volver a leerse si se enrollaba la cinta sobre un bastón idéntico. El escritor Polibio inventó el cuadro de 5 × 5, que se utilizó mucho en diferentes sistemas criptográficos. Julio César empleó un sistema consistente en adelantar cada letra cuatro posiciones.

Una herramienta de esteganografia elaborada en la Universidad Nacional

Bajo la dirección de quien escribe estos comentarios, los estudiantes Pachón, C., Baez, H., Ramos, F., Gómez, D., y Vargas, J. Desarrollaron un programa en que se implementan unas técnicas sencillas de procesamiento de señales para este fin. El lector interesado en el programa puede escribirme al correo electrónico y se lo haré llegar. Una breve descripción de la herramienta se incluye a continuación:

¿Qué es estegol?

Estegol es un programa de esteganografía, es decir, sirve para ocultar información dentro de archivos de imagen bmp de 24 bits. Adicionalmente, permite encriptar la información (hacerla irreconocible mediante contraseñas y algoritmos de cifrado).

Pantalla Principal

Al momento de correr la aplicación Estegol (en un ambiente Windows), el usuario encontrará la pantalla que se ilustra en la Figura1.

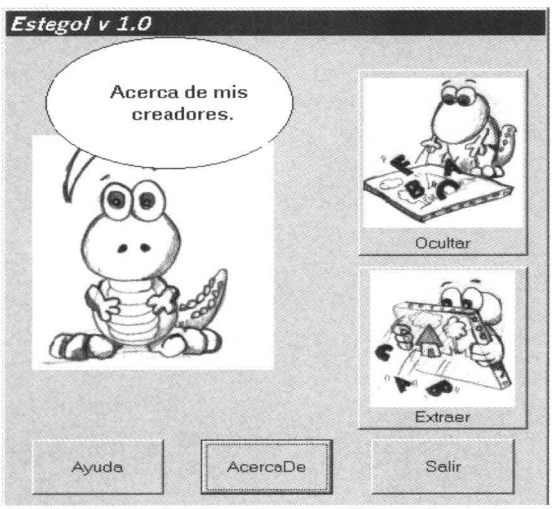

Figura 1. *Pantalla principal de Estegol.*

A medida que el usuario pasa el mouse sobre los botones, en el globo aparecerá información de ¿qué hace el botón?. El botón «Ayuda» traerá información adicional sobre la aplicación y los algoritmos implementados. El botón «Acerca De» muestra los créditos del programa, y salir es para terminar la aplicación.

¿Cómo ocultar información en una imagen?

Lo primero que el usuario debe hacer, es pulsar el botón Ocultar en la pantalla principal (Figura 1). Después de hacer click en este botón, aparecerá un asistente que le guiará paso a paso a través del proceso como lo muestra la figura 2.

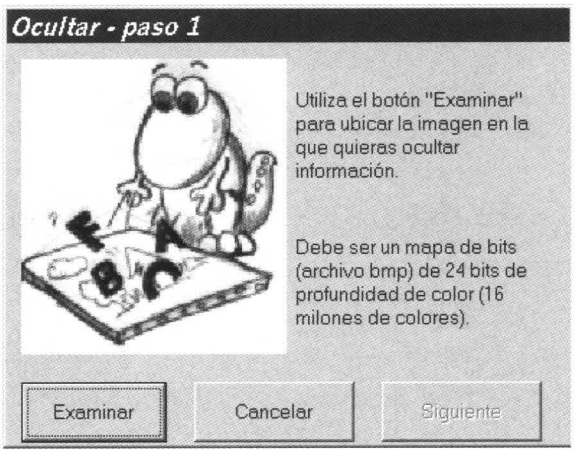

Figura 2. *Ocultar -paso1.*

Aquí es necesario dar click en el botón «Examinar» y ubicar en el disco duro una imagen BMP de 24 bits, en la cual se desea ocultar información. Recuerde que debe ser lo suficientemente grande para poder ocultar un archivo de buen tamaño. Un criterio inicial es escoger una imagen de al menos tres veces el tamaño de lo que se quiera ocultar.

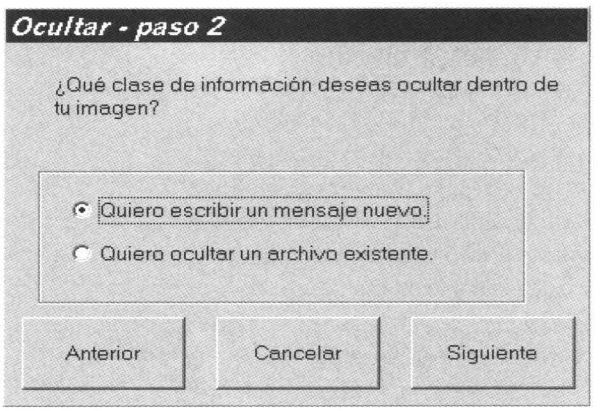

Figura 3. *Ocultar-paso 2.*

Una vez escogida la imagen y hecho click en el botón siguiente, accederá al paso 2 del asistente (Figura 3). Aquí se debe elegir entre esconder un mensaje nuevo o un archivo existente. Si elige mensaje nuevo, aparecerá una caja de texto en la que debe escribir su mensaje, de lo contrario, verá un diálogo parecido al paso 1 en el que es necesario ubicar el archivo a ocultar (Figura 5).

Después de haber elegido lo que se desea ocultar, se debe escoger si se quiere encriptar. Si el usuario decide encriptar, debe indicar con cuál de los dos métodos disponibles: Kabadonga y Espiga. Recuerde que encriptar hace más lento el proceso, pero es más seguro.

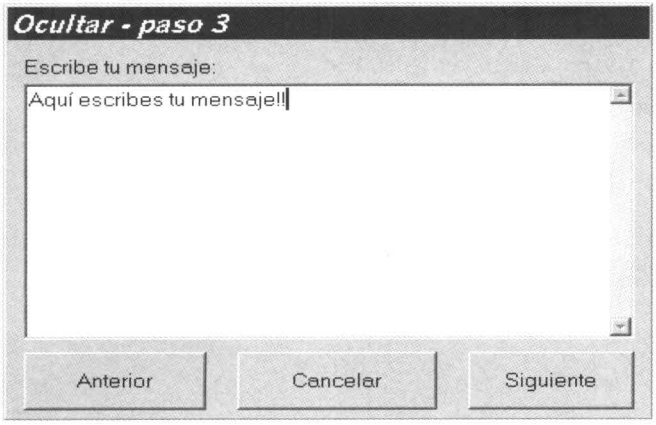

Figura 4. *Ocultar-paso 3.*

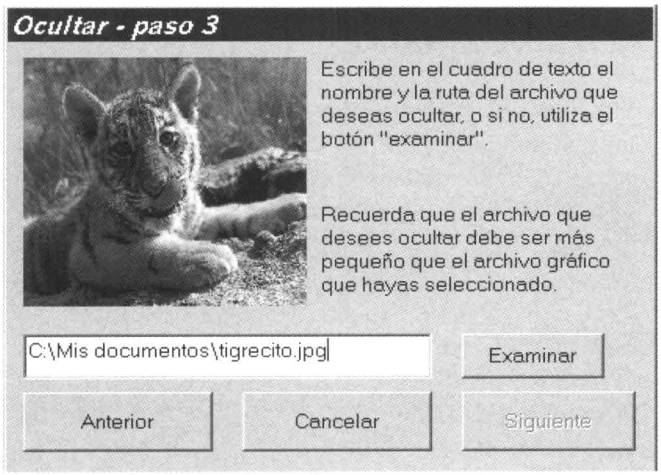

Figura 5. *Ocultar-paso 3 (imagen).*

Si decide no encriptar, al hacer siguiente arrancará una animación que durará mientras el algoritmo termina. Si decide encriptar, antes debe suministrar una contraseña (Figuras 6 y 7).

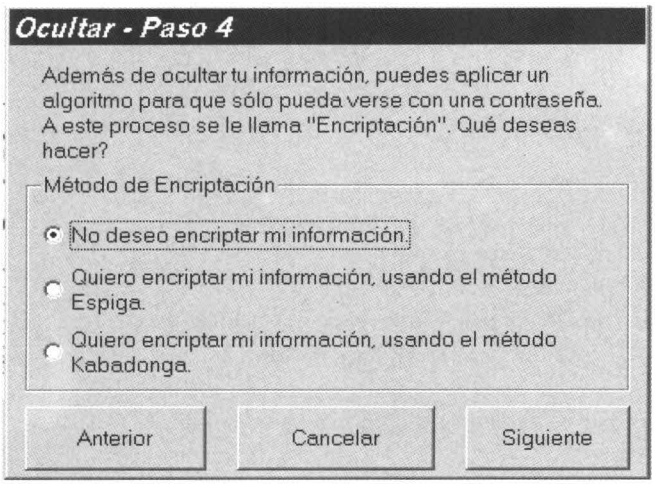

Figura 6. *Ocultar paso 4 (contraseña).*

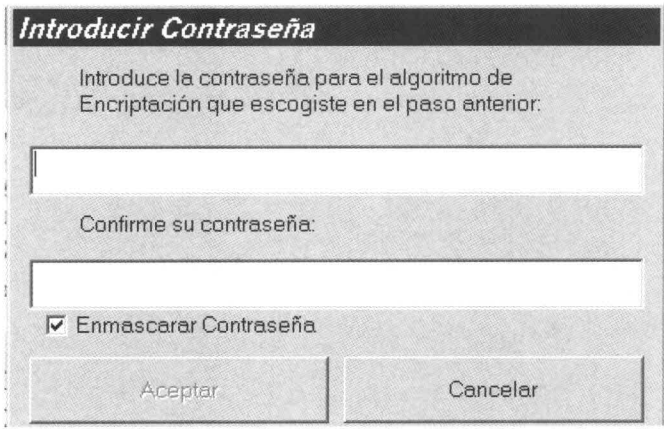

Figura 7. *Introducir contraseña.*

una vez validada la contraseña, pasará a la animación (Figura 8). Cuando termine, el archivo estará listo, y volverá a la pantalla principal.

Figura 8. *Cuadro de animación mientras trabaja el sistema .*

En ese momento la información, aunque no se ve en la imagen, quedará allí almacenada y podrá recuperarse como se indica a continuación.

¿Cómo rescatar información embedida de una imagen?

Para poder sacar información de una imagen, debe tener una imagen en la que se haya ocultado algo. Es necesario saber si está encriptada, y si es así, con qué método y cuál fue la contraseña usada. Para entrar al asistente, debe dar click en el botón sacar de la pantalla principal (Figura 1). A continuación, debe suministrar la ruta del archivo dando click en el botón examinar de la pantalla que se muestra en la figura 9.

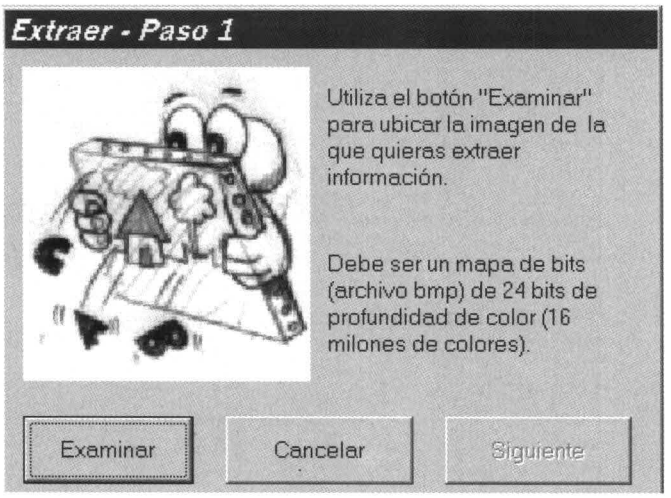

Figura 9. *Extraer - paso 1.*

Recuerde que el archivo debe ser una imagen bmp de 24 bits.

Una vez realizados todos los pasos que el asistente indica, aparecerá una animación que durará el tiempo que demore el programa en extraer la información (Figura 10).

Figura 10. *Animación para extraer información.*

Cuando termine, aparecerá una pantalla informándole el resultado del proceso. Si lo que estaba oculto era un mensaje o una imagen, los podrá ver directamente. Si fue un archivo, verá su nombre, y con el botón «guardar» puede elegir una ruta en disco para almacenarlo. Si no usa el comando guardar como, el archivo será borrado al salir del programa.

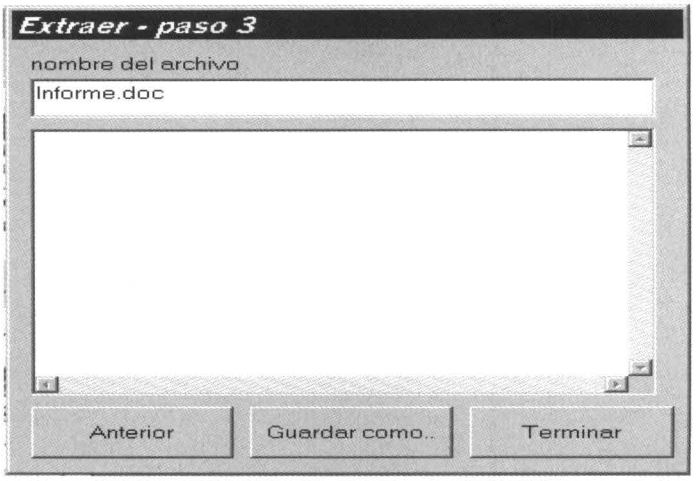

Figura 11. *Extraer - paso 3.*

Puede parecer asombroso pero en realidad lo que se ha implementado es una serie de procedimientos que se enmarcan dentro de la teoría de señales. Brevemente en la sección que sigue se presenta una descripción de los métodos usados. Es de anotar que la mayoría de ellos (y los que por espacio no se presentan aquí) son diseñados en la Universidad Nacional o han sido variantes de técnicas que se trabajan a nivel mundial.

Método de esteganografía

Para comprender cómo funciona la esteganografía, primero es necesario saber algo acerca de cómo se almacena una imagen con formato bmp en el computador. Despúes, puede revisar tanto el método usado para la inmersión como el usado para la recuperación.

¿Cómo está compuesto un archivo BMP?

El archivo BMP contiene dos cabeceras que contienen los siguientes datos:

1. En la cabecera inicial se incluyen:
- El identificador del archivo ("BM").
- El tamaño total del archivo.
- 4 bytes reservados.
- Posición del primer byte del primer píxel, que en este caso debe ser el Byte número 55.

2. En la segunda cabecera, que se encuentra a continuación de la inicial se incluye:
- Tamaño de esta cabecera.
- Anchura del gráfico en bytes.
- Altura del gráfico en bytes.
- Número de planos utilizado.
- Número de bits por píxel (Espectro), que en nuestro caso debe tener un valor de 24.
- Dato que indica si está comprimido, en nuestro caso no debe estarlo (tener un valor de 0).
- Tamaño neto de la imagen (Ancho por Alto).
- Número de bytes por metro en el eje X.
- Número de bytes por metro en el eje Y.
- Número de colores usados.
- Número de colores importantes.

El tamaño total de estas dos cabeceras es de 54 bytes.

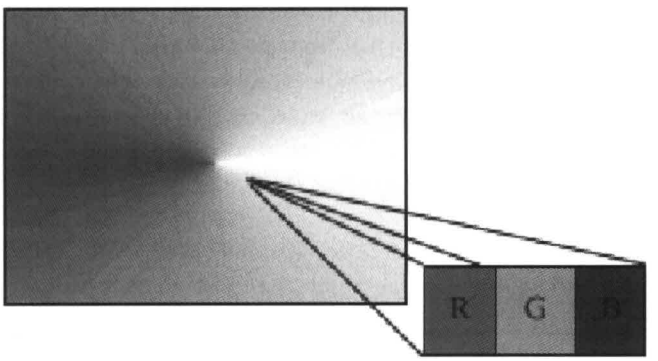

Figura 12. *Sistema RGB*

Un píxel está compuesto por tres canales de color, que son rojo, verde y azul (RGB), y en este caso, cada canal está compuesto por un byte (8 bits), de tal forma que un píxel utiliza 24 bits para representarse (Figura 12).

También es importante saber que la altura del gráfico debe ser múltiplo de 4, y si no lo es, se utilizan bytes de relleno para acomodarla a esta exigencia. Para efectos de la esteganografía, también es posible utilizar estos bytes de relleno para ocultar datos sin temor a dañar el archivo gráfico original BMP.

Método de inmersión

Primero, se debe verificar que el archivo BMP tenga un espectro de 24 bits. Segundo, se debe verificar que el archivo que se vaya a ocultar quepa en el BMP, de tal forma que no se vaya a truncar al momento de realizar la inmersión. Después de haberse verificado lo anterior, se realiza la inmersión de una pseudocabecera que, posteriormente, permitirá recuperar el archivo oculto. Esta pseudocabecera tiene un tamaño de 6 Bytes y contiene la siguiente información:
- Los caracteres de identificación "IE" (2 Bytes): Si existe este identificador, es porque hay un archivo oculto.

-El tamaño del archivo (4 Bytes): Se guarda después del identificador "IE" y, como su nombre lo dice, permite recuperar el tamaño del archivo.

Después de guardarse estos datos, se emplea el Algoritmo de dispersión determinado por el tamaño del archivo oculto. Finalmente, se guarda cada uno de los datos en la posición indicada por el algoritmo anterior.

La inmersión de cada dato, tanto de la pseudocabecera como del archivo, se realiza de la siguiente manera:

1. En el caso del identificador "IE", es la posición del primer píxel (en el caso de un BMP con espectro de 24 bits, éste se encuentra en los Bytes 55, 56 y 57), y utilizamos el Algoritmo de inmersión de un caracter para ocultar el caracter "I", posteriormente, en el siguiente píxel (Bytes 58, 59, 60), se utiliza el mismo algoritmo para ocultar el carácter "E".

2. En el caso del tamaño del archivo, es necesario posicionarse en el tercer píxel (Bytes 61, 62, 63) utilizando el Algoritmo de inmersion del tamaño del archivo, donde se usan 4 pixeles para este dato.

3. Para el caso del primer dato del archivo, se utiliza el séptimo píxel (Bytes 73, 74, 75) y el Algoritmo de inmersion de un caracter, los demás datos se ocultan con dicho algoritmo en la posición indicada por el Algoritmo de dispersión.

Algoritmo de dispersión

Este algoritmo permite saber cada cuántos pixeles existe un caracter del archivo y se calcula a partir del tamaño del archivo y del espacio libre en el archivo gráfico BMP. Con este algoritmo, se asegura que los datos se encuentren uniformemente dispersos en todo el gráfico y así poder distorsionar en menor manera dicho archivo, además, también permite que los datos ocultos estén más seguros que si se guardaran de forma secuencial (un dato en un píxel, el siguiente en el siguiente píxel, y así sucesivamente).

Algoritmo de inmersión de un caracter

Este algoritmo es el más importante para el proceso de inmersión de un archivo y consiste en lo siguiente:

1.Se toma el caracter a sumergir en código ASCII, así por ejemplo, si voy a sumergir un archivo de texto y el caracter actual es una "J", entonces tomo su código ASCII que es 74 (Figura 13).

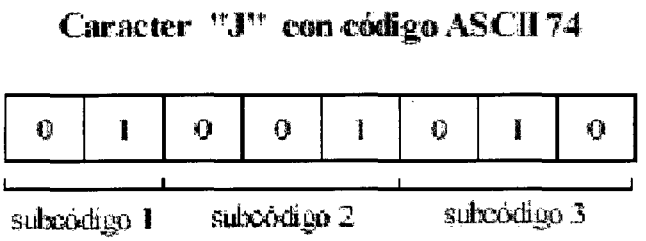

Figura 13. *Código ASCII de la letra J.*

2. Este código se trabaja de forma binaria con longitud de 8 bits y se parte en tres subcódigos, donde dos de éstos son de tres bits y el restante de dos. Así, si tomamos el ejemplo anterior con el código ASCII 74, que en forma binaria es 01001010, obtenemos los tres "subcódigos" (Figura 13).

-El primero se forma con los tres bits de menor valor: 010
-El segundo se forma con los tres bits siguientes: 001
-El tercero y último se forma con los dos bits restantes: 01

3. Después de obtener estos tres subcódigos, se procede a tomar los valores de los tres Bytes que conforman el píxel actual y se pone un valor de 0 en los dos bits de menor valor o peso en el primero de estos bytes, y se pone un valor de 0 en los tres bits de menor valor o peso en los dos bytes restantes. Así, si tomamos un píxel de color azul oscuro con los siguientes valores:

- 153 para el canal azul (primer byte). Equivalente en binario: 10011001
- 51 para el canal verde (segundo byte). Equivalente en binario: 00110011
- 0 para el canal rojo (tercer byte). Equivalente en binario: 00000000

Al volver 0 los últimos valores se tiene:

- 152 para el canal azul (primer byte). Equivalente en binario: 10011000
- 48 para el canal verde (segundo byte). Equivalente en binario: 00110000
- 0 para el canal rojo (tercer byte). Equivalente en binario: 00000000

4. Después en los mismos bits que se pusieron a 0, se sumerge cada uno de los subcódigos del carácter a ocultar (el ejemplo, "J" con ASCII 74), quedando finalmente un píxel de color azul oscuro con los siguientes valores:

- 153 para el canal azul (primer byte). Equivalente en binario: 10011001
- 49 para el canal verde (segundo byte). Equivalente en binario: 00110001
- 2 para el canal rojo (tercer byte). Equivalente en binario: 00000010

De esta manera queda inmerso el dato o caracter en un píxel.

Algoritmo de inmersión del tamaño del archivo

Este algoritmo permite ocultar el tamaño del archivo en el gráfico. Este dato ocupa 4 bytes, y por lo tanto, utilizará 4 píxeles. Este proceso se realiza de la siguiente manera:

1. Puede dividirse el valor del tamaño en cuatro Bytes separados. Así, si el tamaño de nuestro archivo es de 2184 Bytes, este número en binario es: 100010001000 y lo que se hace es dejar cada cifra por aparte así:

- 136 con equivalente en binario de 10001000

- 008 con equivalente en binario de 00001000
- 000 con equivalente en binario de 00000000
- 000 con equivalente en binario de 00000000

2. Realizamos la inmersión de un byte por píxel a través del Algoritmo de inmersion de un caracter.

Método de recuperación

Primero, se debe verificar que el archivo BMP tenga un espectro de 24 bits. Segundo, se debe verificar que exista un archivo oculto mediante el identificador "IE" en los dos primeros pixeles del BMP (primer dato de la "pseudocabecera").

Después de haberse verificado lo anterior, se procede a realizar la recuperación del tamaño del archivo mediante el algoritmo de recuperación del tamaño del archivo. Posteriormente, se recupera el dato de dispersión utilizando el algoritmo de dispersión determinado por el tamaño del archivo oculto. Finalmente, se recupera cada uno de los datos de la posición indicada por el algoritmo anterior.

La recuperación de cada dato, tanto de la pseudocabecera como del archivo, se realiza de la siguiente manera:

1. En el caso del identificador "IE", tomamos la posición del primer píxel (en el caso de un BMP con espectro de 24 bits, éste se encuentra en los Bytes 55, 56 y 57), y utilizamos el Algoritmo de recuperación de un caracter para extraer el caracter "I", posteriormente, en el siguiente píxel (Bytes 58, 59, 60), utilizamos el mismo algoritmo para extraer el caracter "E".

2. En el caso del tamaño del archivo, es necesario posicionarse en el tercer píxel (Bytes 61, 62, 63) y utilizando el Algoritmo de recuperación del tamaño del archivo, donde se utilizan 4 píxeles para este dato.

3. Para el caso del primer dato del archivo, se utiliza el séptimo píxel (bytes 73, 74, 75) y utilizando el Algoritmo de recuperación de un caracter, los demás datos se extraen con dicho algoritmo en la posición indicada por el algoritmo de dispersión.

Algoritmo de dispersión

Este algoritmo permite saber cada cuántos píxeles existe un caracter del archivo, y se calcula a partir del tamaño del archivo y del espacio libre en el archivo gráfico BMP.

Con este algoritmo se asegura que los datos se encuentren uniformemente dispersos en todo el gráfico y así poder distorsionar menor dicho archivo, además, también permite que los datos ocultos estén más seguros que si se guardaran de forma secuencial (un dato en un píxel, el siguiente en el siguiente píxel, y así sucesivamente).

Algoritmo de recuperación de un caracter

Este algoritmo es el más importante para el proceso de recuperación o extracción de un archivo y consiste en lo siguiente:

1. Se toma el primer subcódigo del primer canal del píxel actual, luego el segundo del segundo canal y el tercero del tercer canal.

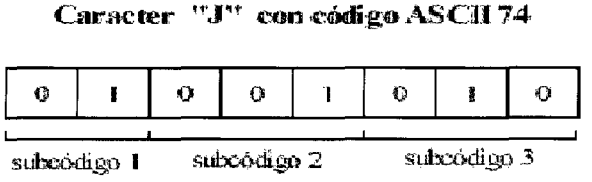

Figura 14. *Caracter J en la extracción.*

2. Finalmente, se unen formando el código ASCII del carácter recuperado, así por ejemplo, si se va a recuperar un archivo de texto y el carácter actual a extraer es una "J" con código ASCII 74 (Figura 14), entonces:

-El primer "subcódigo" a recuperar es: 01
-El segundo "subcódigo" a recuperar es: 001
-El último "subcódigo" a recuperar es: 010

Quedando finalmente el número 01001010 que equivale al ASCII 74 correspondiente a la letra "J". De esta manera queda recuperado el dato o caracter de un píxel.

Algoritmo de recuperación del tamaño del archivo

Este algoritmo permite recuperar el tamaño del archivo en el gráfico. Este dato ocupa 4 Bytes, y por lo tanto utilizará 4 píxeles. Este proceso se realiza de la siguiente manera:

1. Se hace la recuperación de un Byte por píxel a través del Algoritmo de recuperación de un caracter.

2. Se une cada Byte recuperado del tamaño. Así, si por ejemplo, si se recupera:

- 136 con equivalente en binario de 10001000
- 008 con equivalente en binario de 00001000
- 000 con equivalente en binario de 00000000
- 000 con equivalente en binario de 00000000

Obtenemos el número 00000000000000000000100010001000 que equivale al número 2184 que es el tamaño en Bytes.

Conclusiones y recomendaciones

La construcción y mejoramiento de éstas técnicas van a permitir hacia el futuro que las transacciones realizadas por redes como Internet sean más seguras y confiables. En la Universidad Nacional se está trabajando en estos temas de punta en la Tecnología informática con buenos resultados y con herramientas como la que aquí se ha presentado. El lector curioso que simplemente desee emplear el programa para fines prácticos con gusto le suministraremos la herramienta. Pero más allá del simple programa está un área de investigación en las ciencia del computador que es muy prometedora para nuestros ingenieros y profesionales.

REFERENCIAS BIBLIOGRÁFICAS

1. **Ortiz J.** Introducción a la teoría de señales. Universidad Nacional de Colombia. Año 2000. Apuntes de clase.
2. **Oppenheim A.** Señales y Sistemas. Editorial Prentice Hall. 1997.
3. **Pachón C, Baez H, Ramos F, Gómez D, Vargas J.** Manual Técnico Estegol. Universidad Nacional de Colombia. Informe de Curso Año 2001.