RAE

1. TIPO DE DOCUMENTO: Trabajo de grado para optar por el título de INGENIERO DE SONIDO

2. TÍTULO: ALGORITMO PARA LA ENCRIPTACIÓN Y DESENCRIPTACIÓN ENTRE ARCHIVOS DIGITALES DE AUDIO E IMAGEN

3. AUTOR (ES): Luis Eduardo Martínez Bohórquez.

4. LUGAR: Bogotá

5. FECHA: julio de 2017

6. PALABRAS CLAVES: Algoritmo, Encriptación, Audio, Imagen.

7. DESCRIPCIÓN DEL TRABAJO: En el presente proyecto se pretende mejorar los procesos en los cuales se protege la información digital presentada como audios con el fin de que estos archivos no sean de fácil acceso a terceros y puedan ser utilizados de forma inapropiada y contraproducente para su propietario, al mismo tiempo hacer que la perdida de información obvia al momento de modificar un archivo digital sea mínima.

8. LÍNEA DE INVESTIGACIÓN: DISEÑO DE SISTEMAS DE SONIDO

9. METODOLOGÍA: Tiene un enfoque cuantitativo debido a que las conclusiones serán basadas en porcentajes de comparación entre las variables que presenta el proyecto.

10. CONCLUSIONES:

- El algoritmo de encriptación toma como entrada un archivo audio en formato WAV con cualquier frecuencia de muestreo, profundidad de bits, nombre y tamaño; logra encriptarlo en una imagen de formato JPEG que es ilegible a menos que se aplique la desencriptación para obtener nuevamente el audio formato WAV.
- La implementación del algoritmo en el lenguaje JAVA permite que este pueda ser aplicado en diversas plataformas. El resultado de ello es una encriptación completamente funcional, que aumentara su nivel de seguridad al momento de modificar el orden de la información encriptada.
- Al archivo resultante de la encriptación, aunque no presenta pérdida alguna, no es del todo portable, debido a que para ser transferido en medios como WhatsApp sería necesario agregar información de *header* que permita identificar los Streams como una imagen compatible, esto no solo aumentara el tamaño del archivo sino que deberá ser eliminada en la desencriptación, ya que se afecta la reproducción del archivo recuperado al revertir el proceso de cifrado.
- El algoritmo diseñado aplica la forma de encriptación más clásica pero adiciona un plus que es el manejo mucho más avanzado de los formatos de fuente y de destino y es en este punto donde se logra no solo evitar en absoluto las perdidas sino hacer mucho más difícil reconstruir el archivo de audio WAV sin tener la llave indicada, esto debido a que las combinaciones posibles con las muestras tomadas en un solo segundo más los datos del *header* son demasiadas y adicionalmente no cualquier programa permitirá trabajar con el archivo encriptado.

Universidad de San Buenaventura sede Bogotá
Facultad de Ingeniería
Carrera Ingeniería de Sonido

ALGORITMO PARA LA ENCRIPTACIÓN Y DESENCRIPTACIÓN ENTRE ARCHIVOS DIGITALES DE AUDIO E IMAGEN

Tesis para optar el Título profesional de Ingeniero de Sonido.

Luis Eduardo Martínez Bohórquez
Código 20121235060

Asesor

Ingeniero Manuel Torres

Bogotá – Colombia

(Hoja en blanco)

ALGORITMO PARA LA ENCRIPTACIÓN Y DESENCRIPTACIÓN ENTRE ARCHIVOS DIGITALES DE AUDIO E IMAGEN

TABLA DE CONTENIDO

ÍNDICE DE IMAGENES

ÍNDICE DE ANEXOS

INTRODUCCIÓN

Con la masificación de internet a través de los años este ha sido implementado para diversos fines económicos, sociales, políticos etc. Parte importante del uso de la red es el manejo multimedia que facilita el proceso comunicativo y el flujo de la información dentro de la red.

Posteriormente se hizo importante el manejo y almacenamiento de los archivos multimedia en formato digital, creando plataformas como DROPBOX o ICLOUD en donde la información puede ser compilada y consultada fácilmente desde cualquier dispositivo con acceso a red y una cuenta privada para cada usuario.

El objetivo de los llamados "piratas informáticos" es infiltrarse en la red para obtener la información que es resguardada por entidades y personas que puedan significar algún lucro o beneficio personal, este problema hizo importante aplicar diferentes estrategias para que el acceso no fuera tan sencillo y para proteger la información compilada.

Una forma de conseguir esta finalidad es la encriptación, una técnica que data desde el principio de la comunicación misma; es un proceso que consiste en ocultar la información aplicando algún método que permita el hecho de que solamente el emisor y el receptor puedan entender el contenido del mensaje aumentando así la seguridad en el medio que lleva el mensaje.

En el presente proyecto se pretende mejorar los procesos en los cuales se protege la información digital presentada como audios con el fin de que estos archivos no sean de fácil acceso a terceros y puedan ser utilizados de forma inapropiada y contraproducente para su propietario, al mismo tiempo hacer que la perdida de información obvia al momento de modificar un archivo digital sea mínima.

La implementación de esta idea será un algoritmo desarrollado en java que permita encriptar y desencriptar audios e imágenes entre sí, este algoritmo permitirá que tanto el archivo original como el archivo encriptado puedan ser almacenados y visualizados en cualquier dispositivo además de reducir el porcentaje de perdida de información lo máximo posible.

CAPÍTULO I: PLANTEAMIENTO DEL PROBLEMA

1.1 ANTECEDENTES

- Esqueda Elizondo describe los tipos de imágenes que son aceptables para Matlab, además da comandos que se implementan para la transformación entre tipos de imágenes a lo largo de un tratamiento digital de las mismas en procesos de filtrado reducción de ruido e impresión, genera mapas de color e imágenes en escala de grises partiendo de la relación existente entre pixeles y bites de información, como resultado obtiene imágenes de excelente calidad representadas digitalmente como matrices de información que serán empleadas en procesos matemáticos de todo tipo. [1].

- La empresa *Reifi* muestra como apoyado en procesamiento digital de señales crea una imagen 3D que posteriormente imprimé como una representación física de una señal de audio digital, esta impresión 3D es posteriormente escaneada en un dispositivo móvil almacenada y reproducida para obtener el mensaje original, logrando un canal de transmisión moderno y una codificación de mensaje totalmente diferente enfocado más a investigación que a un propósito comercial. [2].

- Los autores Mora Pascual, Azorín López, Mora Mora y Fuster Guilló describen las características de una imagen en formato JPEG con el objetivo de hacerlas mas eficientes al momento de suplir las necesidades de cada aplicación y las condiciones de ancho de banda de red por medio de procesos de compresión, obteniendo como resultado un algoritmo que permite evaluar los términos de relación de compresión, retardos de compresión y calidad luego de comprimir la imagen al ser comparadas con procedimiento estándar. [3].

- Los autores Tian, Sambasivam y Barron crean una imagen digital de una grabación en disco de vinilo utilizando un escáner de resolución óptica de 2400 dpi para posteriormente digitalizar el audio que estaba grabado de forma análoga en el disco, con el objetivo de mejorar la calidad del mismo, después de realizar un tratamiento a la señal ahora digital como filtrado y ecualización; la conclusión a la que llegaron es que se lograría mejorar aún mas la calidad de la imagen y del audio digital mejorando el escáner y la iluminación para reducir el ruido en la señal digital además de reducir costos y procesos dentro del proceso. [4].

- El autor Mendoza Manzano construye un hardware que implementara en el procesamiento y análisis de imágenes con el fin de hacer un tratamiento digital como una mejora en la calidad de la misma, aplicando distintos tipos de filtrado con el fin de eliminar la distorsión de colores; obteniendo como resultado una herramienta

competitiva en el procesamiento y análisis de una imagen digital que además permitiera agregar algoritmos creados propiamente por el usuario que pudiesen mejorar el dispositivo realizado. [5].

- Los autores Neimeyer y Edler diseñaron e implementaron un algoritmo que permitiera detectar además de extraer las transientes en un audio digital codificado con técnicas como mp3 y MPEG con propósitos científicos pero una gran proyección para procesos de mezcla y edición, el resultado es un algoritmo funcional que permite la detección además de la extracción de transientes en un audio digital además de un campo muy grande de trabajo a futuro. [6].

- El autor Rahul R Upadhyay realizo un estudio de encriptación y desencriptación de archivos digitales de audio en formatos de imagen con el objetivo de identificar las características propias de la imagen y analizando cual presenta mayor porcentaje de compresión afectando el archivo que resulta de este proceso, como conclusión obtuvo que el formato JPEG presenta una compresión que puede llegar a comprometer seriamente la calidad del archivo de audio y aconseja finalmente el formato TIFF para realizar este tipo de procesos de encriptación debido a que su forma de codificar datos no los afecta como los otros que examino para el desarrollo de su investigación. [7].

1.2 DESCRIPCIÓN Y FORMULACIÓN DEL PROBLEMA

El manejo de los archivos digitales de audio e imagen tienen diferentes cosas a favor y en contra, con el desarrollo de la tecnología se han venido mejorando aspectos como la calidad en los diferentes tipos de imágenes que manejan los dispositivos electrónicos, otros aspectos que han cobrado relevancia son la portabilidad y la facilidad requerida a la hora de visualizar los archivos; con el desarrollo desmedido que tiene la tecnología han surgido nuevos formatos con características específicas que los hacen más útiles en algunos casos a comparación de otros.

En cuanto a imagen uno de los formato más utilizados es el JPEG el cual es compatible con cualquier dispositivo, es utilizado por cámaras profesionales Smartphone y de más dispositivos con una cámara integrada debido a que un archivo puede llegar a ocupar muy poco espacio en memoria con una calidad sobresaliente. El formato JPEG ***"Joint Photographic Experts Group"*** es el formato admitido a nivel profesional, es de fácil reproducción en cualquier dispositivo y su espacio de memoria en disco no es tan alto; este formato tiene unas características específicas entre las cuales están:

- Muy alto grado de compresión.

- No permite animaciones.
- Numero de colores – 24 bits a color o 8 bits B/N.

En cuanto al audio el desarrollo no se queda atrás, los diferentes formatos son similares entre si aunque sus pequeñas características son a su vez una gran diferencia. En los dispositivos móviles se admiten diferentes formatos dependiendo de lo avanzado que sea y la capacidad de almacenamiento disponible, el formato estándar en el audio profesional es WAV que no brinda precisamente facilidad de almacenamiento debido a que su espacio en memoria de disco es alto. Este formato fue desarrollado por las compañías Microsoft e IBM, y su nombre proviene de las WAVE que significan ***Waveform Audio File Format***, es uno de los formatos que no poseen compresión en los datos por lo que presenta una excelente calidad.

La protección de los archivos que brindan los sistemas operativos e incluso programas diseñados con este propósito consisten en colocar claves para que el acceso sea limitado solo a aquellos usuarios que conocen la combinación de caracteres correcta, otros programas especializados modifican algunas características del archivo seleccionado haciéndolo ilegible para terceros a manera de protección.

La encriptación es uno de los métodos más implementados a la hora de ocultar información importante y privada a terceras personas, las diferentes clases de encriptación varían en la forma de modificar el archivo que se desea proteger. Además dejar unas etiquetas propias de cada tipo de encriptación utilizadas para que el receptor del mensaje sepa cómo proceder para acceder código encriptado.

La idea central que quiere tratar este proyecto es encriptar en formato JPEG un audio en formato WAV y viceversa con el fin de dar una protección adicional a los archivos digitales en estos formatos, asegurando una perdida mínima al momento de encriptar y desencriptar. Adicionalmente realizar este proceso de forma fácil y rápida asegurando también que el archivo encriptado y el archivo resultado de la desencriptación puedan ser transportados en cualquier dispositivo de almacenamiento portable.

La necesidad de desarrollar este tipo de proyectos surge al momento de que la información privada de las personas o corporaciones es robada y utilizada con la finalidad de obtener lucro o simplemente afectar la integridad o los intereses de los mismos, cabe la necesidad de proteger la información de una manera fácil y poco engorrosa además de eso no afectar la facilidad de hoy en día de visualizar archivos en cualquier momento de forma rápida.

¿Es posible proteger la información mediante el procesamiento digital de señales encriptando y desencriptando audios en imágenes y viceversa asegurando una pérdida de información mínima posterior al proceso?

1.3 JUSTIFICACIÓN

Debido a la cantidad de inconvenientes presentados al momento de almacenar información en las diferentes plataformas o dispositivos, además de hacerla vulnerable y asequible a personas que pueden usarla en cualquier propósito sin autorización del usuario afectando la tranquilidad de los usuarios frente a los diferentes medios digitales.

Algunos casos como por ejemplo el de la actriz norteamericana Jennifer Lawrence reportado por el periódico el heraldo publicado el 16 de septiembre del año 2016, según reporta es la tercera vez que “hackers” o también llamados piratas informáticos obtienen fotos intimas y conversaciones de la celebridad sin su consentimiento y las divulgan por diferentes redes sociales afectando su vida personal. [8].

La industria de la música también se ve afectada de forma monetaria por este tipo de hechos el diario El mundo, dos “hackers” alemanes han robado y publicado canciones inéditas de artistas como Lady Gaga y Justin Timberlake entre otros introduciendo un virus informático en sus computadores personales y los computadores de sus sellos discográficos robando estos archivos y vendiéndolos en la internet, causando pérdidas millonarias a sus víctimas. [9].

Como estos muchos otros casos en donde la información digital de archivos de audio e imagen es publicada sin autorización alguna de sus propietarios afectando su vida social, íntima o económica debido al manejo de la seguridad al momento de transportar o almacenar archivos. Por ello se realiza este proyecto de grado, con el fin de brindar una medida de seguridad fácil y rápida a la información personal o corporativa dada en imágenes o audios aplicando procesos de encriptación a estos archivos que los haga irreconocibles a simple vista, además de esto la importancia de preservar una excelente calidad en los archivos al momento de revertir el proceso para que sea acogido por los usuarios de buena manera.

1.4 OBJETIVOS DE INVESTIGACIÓN

1.4.1 OBJETIVO GENERAL

Diseñar e implementar un algoritmo que permita encriptar y desencriptar imágenes de formato JPEG a audio en formato WAV y viceversa.

1.4.2 OBJETIVOS ESPECIFICOS

- Diseñar el algoritmo de encriptación y desencriptación entre los dos formatos.
- Implementar el algoritmo en un lenguaje de programación para verificar su funcionalidad.
- Realizar un protocolo de pruebas del sistema para determinar el porcentaje de perdida de información.

1.5 ALCANCES Y LIMITACIONES DEL PROYECTO

1.5.1 ALCANCES

- Implementarse en diferentes plataformas de almacenamiento como una medida de seguridad adicional brindada a los usuarios para proteger su información.
- Implementarse en diferentes dispositivos móviles popularizando el formato de audio WAV en personas que no manejen el audio profesional.
- Implementarse en otros formatos populares de audio e imagen que sean implementados en diferentes especialidades.
- Describir formatos de video y encontrar una forma de encriptarlos.

1.5.2 LIMITACIONES

- Diferencia la profundidad de bits entre los archivos a encriptar que podrían representar perdidas al momento de recuperar el archivo original.
- El espacio en memoria que puede ocupar el archivo encriptado dentro de un dispositivo de almacenamiento.

CAPÍTULO II: MARCO TEÓRICO

2.1 ENCRIPTACIÓN

La criptografía según la RAE - Criptografía: Arte de escribir con clave secreta o de un modo enigmático.
Es la creación de técnicas para el cifrado de datos, Teniendo como objetivo conseguir la confidencialidad de los mensajes. Si la criptografía es la creación de mecanismos para cifrar datos, el criptoanálisis son los métodos para "romper" estos mecanismos y obtener la información. Una vez que nuestros datos han pasado un proceso criptográfico decimos que la información se encuentra cifrada.
Hasta el 2014 la palabra encriptación era una mala traducción del inglés *encrypt*, la Real Academia Española aceptó los términos encriptar y desencriptar, por lo que se considera válido usarlos en la actualidad.
Existen diferentes técnicas de criptografía una de las más antiguas y las más comunes es la criptografía simétrica que consiste en realizar siempre el mismo tipo de cifrado o encriptación generando una clave o método para revertir el proceso (puede entregarse al receptor en cualquier de cualquier manera), uno de los grandes beneficios es la velocidad con la que puede llegar a realizarse que la hace apropiada a la hora de encriptar grandes cantidades de datos y transportarlos hasta el receptor del mensaje; al mismo tiempo la desventaja más grande es la necesidad de distribuir la clave debido a que si algún tercero llegase a hacerse con la clave y el mensaje el riesgo de revelar la información sería muy alto. [10]

2.1.1 CRIPTOGRAFÍA SIMÉTRICA: DES Y AES

La criptografía Simétrica es un método criptográfico, usa la misma clave para cifrar y descifrar. Esto supone un grave problema a la hora de realizar el intercambio entre el emisor y el receptor, dado que si una tercera persona estuviese escuchando el canal podría hacerse con la clave, siendo inútil el cifrado. Es importante que la clave sea difícil de adivinar y el método de cifrado empleado el adecuado. Hoy en día, con la capacidad computacional disponible, si se emplean los algoritmos adecuados, dependiendo del método de cifrado empleado se puede obtener una clave fácilmente.

DES:

El algoritmo DES es un algoritmo de cifrado desarrollado por la NSA a petición del gobierno de EEUU bajo la presión de las empresas por la necesidad de un método para proteger sus comunicaciones.

DES es un algoritmo de cifrado por bloques. Toma un bloque de una longitud fija de bits y lo transforma mediante una serie de operaciones básicas en otro bloque cifrado de la misma longitud. En el caso de DES el tamaño del bloque es de 64 bits. La clave también tiene 64 bits pero 8 de estos bits se emplean para comprobar la paridad, haciendo que la longitud efectiva de la clave sea de 56 bits.
DES se compone de 16 fases o rondas idénticas. Al comienzo y al final se realiza una permutación. Estas permutaciones no son significativas a nivel criptográfico, pues se incluyeron para facilitar la carga y descarga de los bloque en el hardware de los años 70. Antes de cada ronda el bloque se divide en dos mitades de 32 bits y se procesan alternativamente. Este proceso es conocido como esquema Feistel. Este nos proporciona un proceso de cifrado y descifrado casi iguales. La única diferencia es que las sub-claves se aplican de forma inversa al descifrar. [11]

Función F-Feistel:

1. Expansión: se toma la mitad del bloque de 64 bits (32bits) que son expandidos a 48 bits mediante la permutación de expansión, denominada E en el diagrama, duplicando algunos de los bits.
2. Mezcla: el resultado se combina con una subclave utilizando una operación XOR. Dieciséis subclaves (una para cada ronda) se derivan de la clave inicial mediante la generación de subclaves.
3. Sustitución: tras la mezcla, el bloque es dividido en ocho trozos de 6 bits que se pasan a las cajas de sustitución. Cada una de las ocho S-cajas reemplaza sus seis bits de entrada con cuatro bits de salida, de acuerdo con una trasformación no lineal, especificada por una tabla. Las S-cajas constituyen el núcleo de la seguridad de DES, sin ellas, el cifrado sería lineal y fácil de romper.
4. Permutación: finalmente, los 32bits salientes de las S-cajas se reordenan de acuerdo a una permutación fija.

Generación de claves:

De los 64 bits iniciales se toman 56 con la Elección Permutada 1 (PC-1). Estos 56 bits son divididos en dos mitades de 28 bits que serán tratadas de forma independiente. En rondas sucesivas se desplazan los bits de ambas mitades 1 o 2 bits a la derecha. Tras el desplazamiento se toman 48 bits (24+24) mediante la Elección Permutada 2 (PC-2). Al realizar un desplazamiento en cada ronda cada subclave estará empleando un conjunto diferente de bits.
La generación de claves para descifrado es similar, la única variación es que se deben generar en orden inverso.

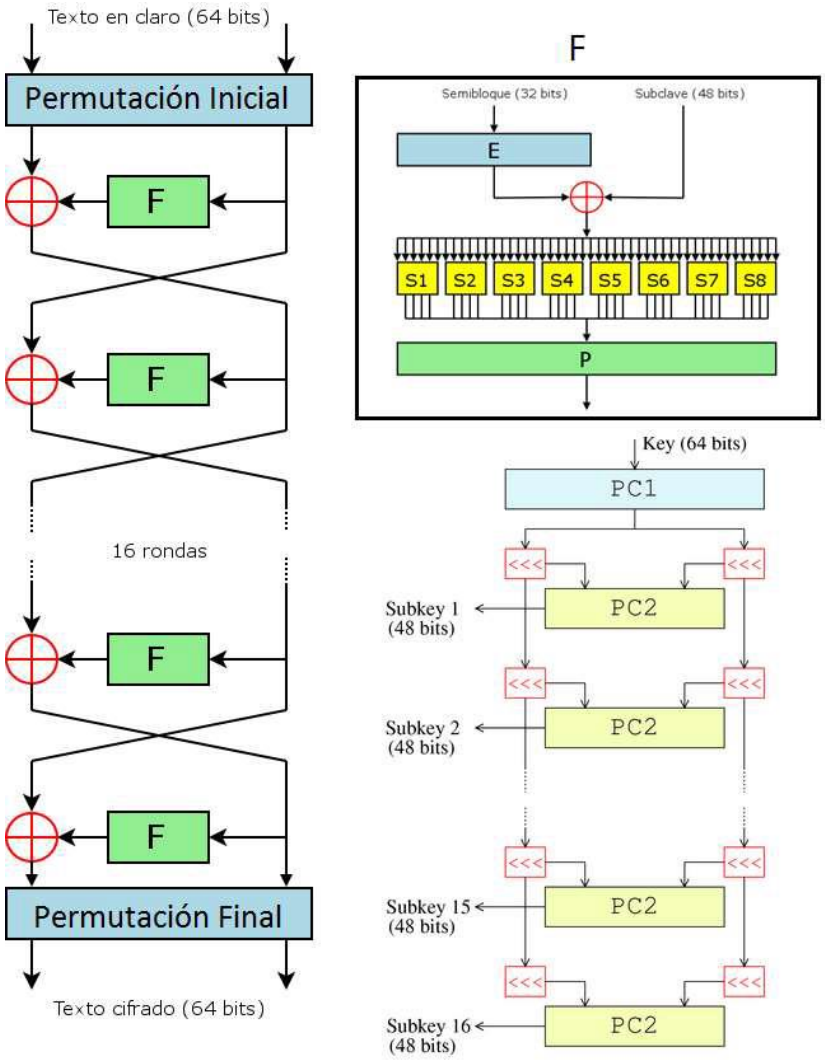


Imagen 1. Cifrado y descifrado DES
Imagen obtenida de http://www3.uah.es

AES:

El algoritmo AES (Advanced Encryption Standard) también conocido como Rijndael opera sobre una matriz de 4x4 bytes. Mediante un algoritmo se reordenan los distintos bytes de la matriz. El cifrado es de clave simétrica, por lo que la misma clave aplicada en el cifrado se aplica para el descifrado.
Basado en El algoritmo Rijndael, Al contrario que su predecesor DES, Rijndael es una red de sustitución permutación, no una red de Feistel. AES es rápido tanto en software como en hardware, es relativamente fácil de implementar, y requiere poca memoria.
El algoritmo AES funciona mediante una serie de bucles que se repiten. 10 ciclos para claves de 128 bits, 12 para 192 y 14 para 256. [11]

Supongamos que tenemos 2 matrices: matriz a y matriz k.

$$a00\ a01\ a02\ a03$$
$$a10\ a11\ a12\ a13$$
$$a20\ a21\ a22\ a23$$
$$a30\ a31\ a32\ a33$$

$$k00\ k01\ k02\ k03$$
$$k10\ k11\ k12\ k13$$
$$k20\ k21\ k22\ k23$$
$$k30\ k31\ k32\ k33$$

En la matriz a tenemos nuestra información y en la matriz k tenemos una subclave generada a partir de la principal.

El algoritmo de cifrado es el siguiente:

1. Expansión de la clave. Mediante una serie de operaciones se obtienen n+1subclaves a partir de la clave principal (n es el número de ciclos).
2. Ciclo inicial. AddRoundKey. Se aplica una XOR byte byte entre la matriz a y la matriz k.
3. Ciclos intermedios.

   3.1 SubBytes. Tomando como referencia una tabla especificada cada byte es sustituido por otro en función de la tabla.

   3.2 ShiftRows. Cada byte de cada fila es desplazada n-1 huecos a la izquierda (siendo n el número de fila).

   3.3 MixColumns. Los 4 bytes de una columna se combinan entre sí para obtener 4 bytes diferentes. Este proceso se logra multiplicando la columna por una matriz dada.

$$2\ 3\ 1\ 1$$
$$1\ 2\ 3\ 1$$
$$1\ 1\ 2\ 3$$
$$3\ 1\ 1\ 2$$

   3.4 AddRoundKey.

   4 Ciclo final.

   4.1. SubBytes.

   4.2. ShiftRows

   4.3. AddRoundKey

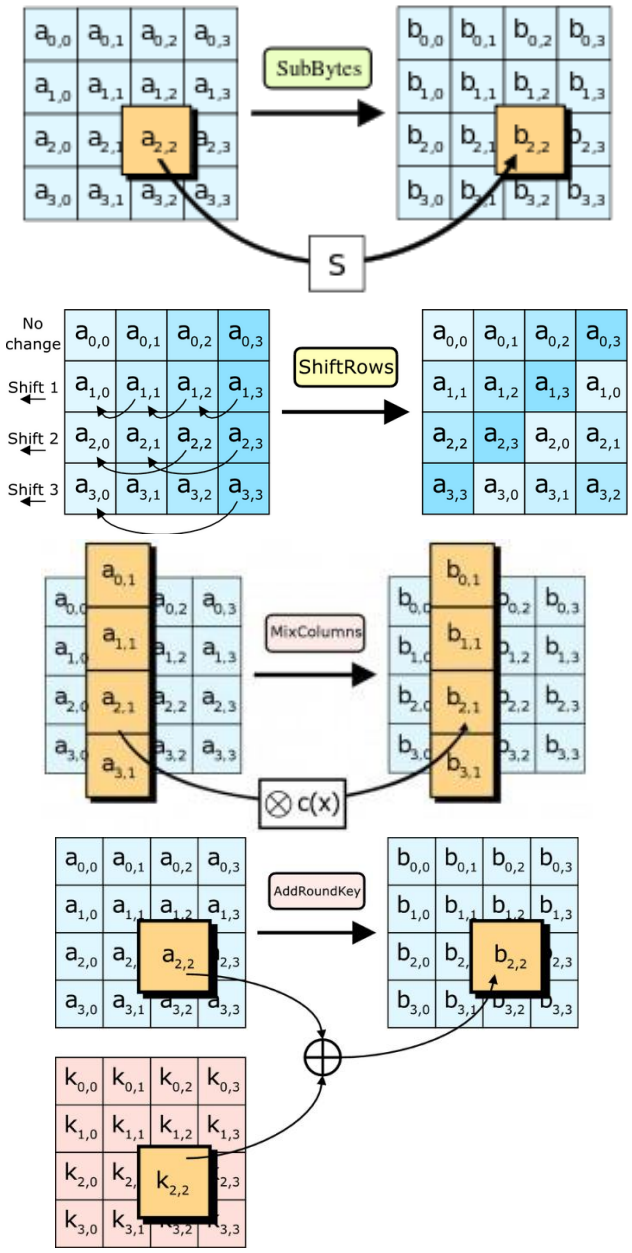


Imagen 2. Encriptación AES
Imagen tomada de http://www3.uah.es

2.1.2 ONE-TIME PAD:

One-time pad es un tipo de algoritmo de cifrado por el que el texto en claro se combina con una clave aleatoria o «libreta» de la misma longitud y que sólo se utiliza una vez. Fue inventado en 1917. Si la clave es verdaderamente aleatoria, nunca se reutiliza y, por supuesto, se mantiene en secreto, se puede demostrar que el método de la libreta de un solo uso es irrompible.

Proviene del cifrado de Vernam. El sistema de Vernam es un cifrado que combina un mensaje con una clave que se lee de un bucle de cinta de papel. En su forma original, el sistema de

Vernam era vulnerable porque la clave se podía reutilizar. El uso único vino un poco después, cuando Joseph Mauborgne reconoció que si la cinta de la clave era completamente aleatoria, se incrementaría la dificultad criptoanalítica. [10]

De acuerdo con Alfred Menezes en su libro, Handbook of Applied Cryptography (Manual de criptografía aplicada), se puede decir que un sistema es totalmente secreto, o incondicionalmente seguro, cuando el texto cifrado que se observa no proporciona información complementaria acerca de la cadena de texto en claro original. Si suponemos que L es el número de bits en la cadena de texto en claro, i va de 1 a L en las siguientes definiciones [11]:

- pi = el bit n° i en la cadena de texto en claro
- ci = el bit n° i en la cadena de texto cifrado
- ki = el bit n° i en la cadena de clave
- P(pi) = la probabilidad de que pi se ha enviado
- P(pi | ci) = la probabilidad de que pi se ha enviado dado que ci se ha observado

Se puede decir que un sistema es perfectamente secreto cuando P (pi) = P (pi | ci). En los sistemas de cifrado de flujo tradicionales, el método más común de mezclar bits de datos de texto en claro con bits de clave es a través de la operación XOR en los bits correspondientes. El hecho de que la clave sea completamente aleatoria nos lleva a varias conclusiones. La probabilidad de observar un bit de la clave es la misma que la de cualquier otro bit y el hecho de conocer valores previos de la clave no nos aporta nada sobre valores futuros.
Enunciando otra definición,

- P(ki=1) = P(ki=0) = 1/2 para todos los i.

Dicho de otra forma, el valor de un bit cualquiera es probablemente 0 o 1.
Conocidos dos elementos cualquiera de {pi, ci, ki} podemos determinar el tercero. Asimismo, conociendo uno entre {pi, ci, ki}, se puede escribir un segundo en relación con el tercero.
Por ejemplo, P (ci=1 | ki=0) = P (pi=1); dicho de otra manera, si sabemos que el bit clave es 0, entonces el texto en claro y el texto cifrado deben ser iguales. Para mostrar que P(pi | ci) = P(pi), primero hay que mostrar que P(ci) = P(ci | pi).
Algunos inconvenientes:

- Requiere libretas de un solo uso perfectamente aleatorias.
- La generación e intercambio de las libretas de un solo uso tiene que ser segura, y la libreta tiene que ser al menos tan larga como el mensaje.
- Hace falta un tratamiento cuidadoso para asegurarse de que siempre permanecerán en secreto para cualquier adversario, y es necesario deshacerse de ellas correctamente para evitar cualquier reutilización parcial o completa —de ahí el «un solo uso».

2.1.3 CIFRADORES DE FLUJO Y CIFRADORES DE BLOQUE: WEP (RC4) Y ECB

Los cifradores de bloque trabajan con clave simétrica sobre grupos de bits de una determinada longitud fija (los bloques). El cifrado de bloque toma en la entrada un bloque de texto plano y produce un bloque de texto cifrado de igual longitud. Esta transformación de texto plano a cifrado se controla mediante una clave secreta. Para el camino inverso (texto cifrado -> texto plano) se opera de la misma manera.
Los cifradores de bloques tienen un inconveniente, hay ciertas aplicaciones que no pueden hacer uso de esta utilidad por estar formadas por un flujo constante de bits. Tenemos por ejemplo un enlace de radio, telefonía, etc. Para estas aplicaciones surgen los cifradores de flujo.
Un cifrador de flujo consiste en un algoritmo que convierte el texto claro en texto cifrado pero trabajando bit a bit. El funcionamiento es similar al de bloque, en la entrada tenemos el flujo de datos. Se genera un "flujo de clave" y la salida es una XOR bit a bit del flujo de datos y el de clave.
Vamos a tratar 2 algoritmos: WEP-RC4 (flujo) y ECB (bloque)

WEP:

El algoritmo WEP (Wired Equivalent Privacy) es el sistema de cifrado incluido en el estándar IEEE 802.1. WEP está basado en el algoritmo de cifrado RC4, teniendo dos variantes, una que usa clave de 64 bits (40 bits más 24 bits de vector de iniciación) y otra que usa clave de 128 bits (104 bits y 24 bits de vector de iniciación). En 2003 la Wi-Fi Alliance anunció la sustitución de WEP por WPA y en 2004 se ratificó el estándar 802.11i (WPA2) declarando WEP-40 y WEP-104 como inseguros. A pesar de ello todavía hoy en día se sigue utilizando.
Como hemos dicho WEP utiliza el algoritmo de cifrado RC4. El funcionamiento de RC4 es el siguiente:
Se expande una semilla (seed) para generar una secuencia de números pseudoaleatorios de mayor tamaño. Posteriormente, se "unifican" el mensaje y la secuencia pseudoaleatoria mediante una función XOR. El problema que presenta este método es que si se utiliza la misma semilla para cifrar dos mensajes diferentes obtener la clave a partir de los dos textos cifrados sería trivial (por trivial entendemos sencillo desde el punto de vista matemático). En un intento de evitar esto, en WEP se incluyó un vector de iniciación de 24 bits que se modifica regularmente y se concatena a la contraseña para generar la semilla.
El principal defecto de WEP se encuentra precisamente en este vector de iniciación. El tamaño del vector de iniciación es constante (24 bits), esto nos da un número limitado de vectores (2^24=16.777.216). El problema es que la cantidad de tramas que pasan a través de un punto de acceso son muy grandes y es fácil encontrar 2 mensajes con el mismo vector haciendo relativamente fácil obtener la clave. Se puede aumentar el tamaño de la clave, pero esto solo incrementará el tiempo necesario para romper el cifrado. [10]

ECB:

ECB (electronic codebook) es uno de los múltiples algoritmos de cifrado de bloque y uno de los más sencillos.
En ECB la información se divide en bloques y cada uno de ellos es cifrado por separado aplicando una clave común. Este sistema implica un problema, con bloques idénticos tendremos bloques cifrados idénticos pudiéndose reconocer así un patrón que facilitará la obtención del mensaje original.
Una mejora de este algoritmo es CBC (cipher-block chaining). Esta mejora consiste en hacer una XOR del bloque antes del cifrado con el anterior cifrado, y posteriormente cifrarlo. Otras evoluciones como CFB y OFB hacen que el cifrado pase a operar como un cifrador de flujo. [10]

2.1.4 CIFRADO ASIMÉTRICO: RSA Y DIFFIE HELLMAN

La criptografía asimétrica (también conocida como de clave pública) es un sistema que emplea una pareja de claves. Esta pareja de claves pertenecen a la misma persona. Una es de dominio público y cualquiera puede tenerla y la otra es privada. El funcionamiento de este sistema es el siguiente:
El remitente usa la clave pública del destinatario y sólo con la clave privada se podrá descifrar el mensaje. De esta forma se consigue que sólo el destinatario pueda acceder a la información.
De la misma forma si el propietario usa su clave privada para cifrar un mensaje sólo se podrá descifrar con la clave pública. Pero, si todo el mundo puede tener acceso a la clave pública ¿Que utilidad tiene esto? Precisamente por esto es interesante el sistema. Usando tu clave privada estas demostrando tu identidad, pues, en teoría, solo tú eres poseedor de esa clave privada.
La mayor ventaja de este sistema es que la distribución de claves es más fácil y segura que usando clave simétrica.
Sin embargo este sistema tiene varias desventajas:

- Mayor tiempo de proceso en mismas condiciones respecto a clave simétrica.
- Claves más grandes en en sistemas simétricos.
- El mensaje cifrado es más grande que el original. Por estas razones el principal uso del cifrado asimétrico es solventar los problemas a la hora de intercambiar las claves del cifrado simétrico. De hecho, normalmente lo que se hace es compartir las claves simétricas mediante cifrado asimétrico para posteriormente pasar a un cifrado simétrico más rápido y menos costoso.

Hablaremos de 2 algoritmos: RSA y Diffie Hellman
RSA:

RSA (Rivest, Shamir y Adleman) es un algoritmo de cifrado asimétrico desarrollado en el año 1977 por los anteriormente citados.
Este algoritmo se basa en escoger 2 números primos grandes elegidos de forma aleatoria y mantenidos en secreto. La principal ventaja de este algoritmo desde el punto de vista de seguridad radica en la dificultad a la hora de factorizar números grandes. RSA es seguro hasta la fecha. La idea del algoritmo es la siguiente:
Tenemos un mensaje M. Empleando un protocolo reversible conocido como patrón de relleno convertimos el mensaje M en un número m menor que otro número dado n. Se genera el mensaje cifrado c:

$$C = M^e \ mod \ n$$

Se obtiene m descifrando el mensaje cifrado c:

$$C = M^e \ mod \ n$$

Generación de claves:

1. Tomamos 2 números primos p y q. Estos tienen que ser aleatorios e impredecibles. Importante impredecibles, porque un proceso puede ser perfectamente aleatorio, pero si se conoce, se puede predecir los valores, y por tanto, resultaría en una baja seguridad.
2. Calculamos n=p*q.
3. Calculamos φ(n). La función φ de Euler se define como el número de enteros positivos menores o iguales a n y coprimos con n (dos números son coprimos si no tienen ningún divisor común distinto 1 o -1). La función tiene las siguientes propiedades:

- φ (1) =1.
- φ(p)= p-1 si p es primo.
- φ(p^k) = (p-1) *p^(k-1) si p es primo y k un número natural.
- φ(m*n) =φ(m)φ(n) si m y n son primos.

De esta forma nos queda que para nuestro n=p*q:

φ(n)=(p-1) *(q-1)

4. -Escogemos un número e menor que φ(n) y que sea coprimo con φ(n). Este número será dado a conocer como exponente de la clave pública.
5. Obtenemos un número d mediante aritmética modular tal que d = e^(-1) mod(φ(n)) o lo que es lo mismo (d*e)-1 tiene que ser divisible por φ(n).

De esta forma tenemos la clave pública formada por (n, e) y la privada formada por (n, d).

p = 61 1º nº	primo privado
q = 53 2º nº	primo privado
n = p*q = 3233	producto p×q
e = 17	exponente público
d = 2753	exponente privado

La clave pública (e, n). La clave privada es (d, n). La función de cifrado e

$$encrypt\ (m) = m^e \pmod n = m^{17} \pmod{3233}$$

Donde m es el texto sin cifrar. La función de descifrado es:

$$decrypt\ (c) = c^d \pmod n = c^{2753} \pmod{3233}$$

Donde c es el texto cifrado. Para cifrar el valor del texto sin cifrar 123, nosotros calculamos:

$$encrypt(123) = 123^{17} \pmod{3233} = 855$$

Para descifrar el valor del texto cifrado, nosotros calculamos:

$$decrypt\ (855) = 855^{2753} \pmod{3233} = 123$$

Como hemos mencionado anteriormente, RSA requiere de un esquema de rellenado dado que sino M puede conducirnos a textos cifrados inseguros. Hay múltiples algoritmos de rellenado, entre ellos podemos destacar OAEP (Optimal Asymetric Encription Padding) o SAEP (Simplified Asymetric Encryption Padding). [10]

Diffie Hellman:

El protocolo de cifrado Diffie-Hellman (recibe el nombre de sus creadores) es un sistema de intercambio de claves entre partes, que no han contactado previamente, a través de un canal inseguro y sin autenticación.
Este protocolo se utiliza principalmente para intercambiar claves simétricas de forma segura para posteriormente pasar a utilizar un cifrado simétrico, menos costoso que el asimétrico. Se parte de la idea de que dos interlocutores pueden generar de forma conjunta una clave sin que esta sea comprometida.

1. Se escoge un número primo p y un generador g que será coprimo de p. Estos 2 números son públicos.
2. Escogemos un número a menor que p y calculamos A = g^a mod p. Enviamos A, p y g al otro interlocutor.
3. El otro interlocutor escoge un número b menor que p y calcula B = g^b mod p. Nos envia B.

Ahora, ambos podemos calcular K= g^(a-b) mod p.
Para nosotros B^a mod p = K y para nuestro interlocutor A^b mod p =K. Usamos K como calve.

Al ser p y g públicos cualquier atacante puede conocerlos. Esto no supone una vulnerabilidad. Aunque el atacante conociera estos dos números y capturase A y B, le resultaría computacionalmente imposible obtener a y/o b y consecuentemente K.
Tomamos p=23 y g=5. Elegimos a=6 y b=15. A=8 y B=19.

$$(g^a \bmod p) = 8 \rightarrow (5^a \bmod 23) = 8$$

$$(g^b \bmod p) = 19 \rightarrow (5^b \bmod 23) = 19$$

Partiendo de estas ecuaciones obtener a y b es un problema conocido como logaritmo discreto.

$$a = \log_{disc_g}(g^a \bmod p) = \log_{disc_5}(8)$$
$$b = \log_{disc_g}(g^b \bmod p) = \log_{disc_5}(19)$$

En este caso si podríamos obtenerlos, pues sabiendo que p=23 y que a y b son menores que p solo tendríamos que probar 22 números diferentes. En la realidad se utilizan números primos del orden de 10^{200} haciendo computacionalmente imposible la resolución.

Este protocolo es vulnerable a ataque man-in-the-middle. Estos ataques consisten en que un tercero se coloca en medio del canal y hace creer a ambos que es el otro. De esta forma se podría acordar una clave con cada parte y servir de "enlace" entre los dos participantes. Para que este ataque sea funcional se necesita saber que método de cifrado simétrico se va a emplear. Ocultar el algoritmo de cifrado no cumple con el principio de Kerkckhoffs de que la efectividad de un sistema no debe depender de que su diseño permanezca en secreto por lo que conocer el sistema que se va a emplear se hace trivial. Para evitar esto se puede emplear un protocolo de autenticación de las partes mediante por ejemplo TLS. [10]

2.1.5 FUNCIONES HASH CRIPTOGRÁFICAS

Una función hash criptográfica es aquella función hash que se emplea en el área de la criptografía. Una función Hash es un algoritmo matemático que, con una entrada A, nos da una salida B. Una función hash puede ser cualquier algoritmo matemático pero tiene que cumplir una serie de propiedades.

1. Para una misma función hash, sea cual sea la longitud de la entrada A la salida B tiene que ser de un tamaño fijo.
2. Para cada A, B tiene que ser única.
3. Tiene que ser rápido y fácil de calcular.
4. No se puede volver a A desde B.
5. No puede presentar colisiones. Esto quiere decir que para dos A diferentes no se puede dar un mismo B. Observando las condiciones 1 y 2 vemos que esto es imposible. La función MD5 nos da como resultado un hash de 128 bits. Esto quiere decir que como máximo hay solo 2^{128} textos diferentes, lo cual, es falso. Se hace por tanto muy importante que las colisiones sean mínimas y que encontrarlas sea muy difícil.

En el caso de funciones hash criptográficas se requiere de forma adicional que sean uniformes (para una A elegida aleatoriamente todos los valores hash son equiprobables) y con efecto avalancha (un cambio de un único bit en A supone una B completamente diferente).
Podemos distinguir dos grupos de funciones: las que tienen como objetivo mantener la integridad de los mensajes (detección de modificaciones) y las que tienen como objetivo verificar el origen del mensaje (autenticación).

MD5:

MD5 es un algoritmo de reducción criptográfico de 128 bits ampliamente usado. El algoritmo consta de 5 pasos:

1. Adición de bits. El mensaje es extendido de forma que su longitud menos 448 sea múltiplo de 512. Este paso se realiza aunque la longitud ya sea congruente con 448 modulo 512.
2. Longitud del mensaje. Un entero de 64 bits que represente la longitud del mensaje antes de la adición se concatena al final del resultado del paso anterior. Si la longitud del mensaje es de más de 64 bits se usan sólo los 64 primeros bits. Tras este paso el mensaje es múltiplo exacto de 512. A su vez, el mensaje es múltiplo de 16.
3. Iniciar el búfer MD. Un búfer de cuatro palabras se inicia con unos valores determinados. Cada palabra A,B,C y D tiene 32 bits. Las palabras tienen los siguientes valores:
   A: 01 23 45 67 B: 89 ab cd ef C: fe dc ba 98 D: 76 54 32 10
4. Procesado el mensaje en bloques de 16 palabras. Tomamos cuatro funciones auxiliares cuya entrada es de tres palabra y su salida es una:

$$F(B, C, D) = (B \wedge C) \vee (\neg B \wedge D)$$
$$G(B, C, D) = (B \wedge D) \vee (C \wedge \neg D)$$
$$H(B,C,D) = B \oplus C \oplus D$$
$$I(B,C,D) = C \oplus (B \vee \neg D)$$

   $\oplus$, $\vee$, $\wedge$, $\neg$ representan las operaciones XOR, AND, OR y NOT.

5. Tras realizar 16 ciclos en los que se efectuan una serie de operaciones con las funciones anteriores sobre B, C y D obtenemos la salida ABCD. [12]

Como estos algoritmos existen muchos otros que han sido diseñados y evolucionados dependiendo también el desarrollo tecnológico y la finalidad con la que necesiten ser diseñados. Para este proyecto será aplicado un algoritmo de cifrado o encriptación simétrica tomando una característica del algoritmo AES que es el hecho de desordenar los datos para aplicar el cifrado, adicionalmente utilizar las características propias del formato original como medida de protección de la información ya que si al modificarse el orden original de los datos de tal forma que la cantidad de combinaciones posibles en el orden de los datos sea tan grande será imposible reconocer los datos que allí reposan haciendo que la única forma de revertir el proceso sea con el archivo que desencripta los datos.

2.2 FORMATO DE ARCHIVOS DE INTERCAMBIO DE RECURSOS (RIFF)

RIFF (Resourse Interchange File Format) es una estructura de archivos etiquetados para archivos de recursos multimedia, RIFF no es un formato de archivo, sino una estructura de

archivos que define una clase de formatos de archivo más específicas. fue creado por Microsoft Corporation e IBM en 1991 y publicados en *la referencia del programador de Microsoft Windows Multimedia*. En otras palabras el empaquetado RIFF es un trozo de información que describe formatos de audio e imagen como WAV y AVI entre otros dándoles al inicio una metadata específica que los caracteriza al momento de su codificación y se divide en segmentos llamados ***chunks*** que diferencian aspectos como el formato el número de canales (audio) la cantidad de bits y de más. [13].

2.3 FORMATO DE ARCHIVO DE SONIDO PCM WAVE

El formato WAVE o WAV es un subconjunto de las especificaciones RIFF de Microsoft para el almacenamiento de archivos multimedia, el archivo RIFF comienza con un encabezado de archivo seguido de una secuencia de fragmentos de datos. Un archivo de onda esta descrito como un trozo único de "datos" que contienen los valores reales de la muestra y están antecedidos por dos sub-bloques que caracterizan el formato. En la siguiente imagen se explicara detalladamente la composición de un archivo de formato WAV. [14]

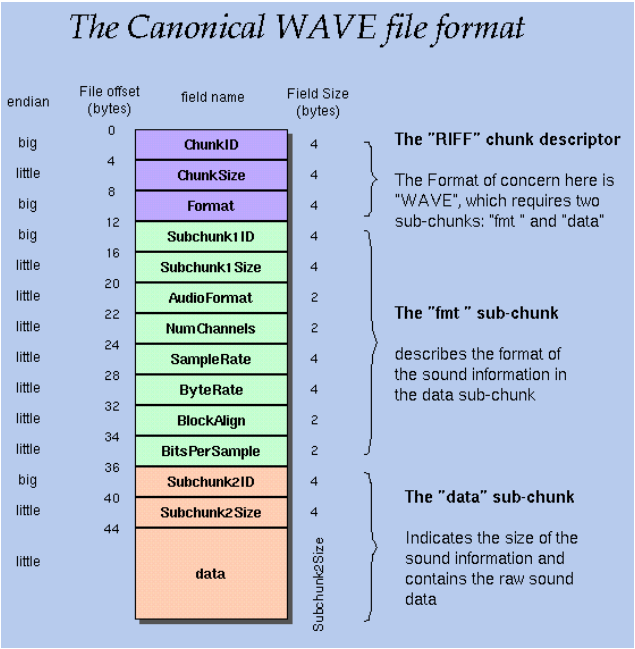


Figura 3. Descripción composición del formato WAV
Imagen tomada de http://soundfile.sapp.org

En los primeros 16 bits de datos se encuentra la cabecera RIFF:

- ChunkID: Contiene las letras "RIFF" en formato ASCII
- Chunksize: Este es el tamaño de la totalidad del archivo en bytes menos de 8 bytes para los dos campos no incluidos en este recuento *ChunkID* y *ChunkSize*.

- Formato: Contiene las letras “Wave”.

Además el formato WAVE o WAV se compone de dos subchunks el “FMT” y “datos”. El “FMT” subchunk describe el formato de los datos de sonido:

- Subchunk1ID: Contiene las letras “FMT”.
- Subchunk1Size: 16 para PCM. Este es el tamaño del resto del Subchunk que sigue este número.
- AudioFormat: PCM=1 es decir la cuantificación lineal (los valores distintos a 1 indican alguna forma de compresión)
- numChannels: Indica el número de canales del audio mono=1, estéreo=2, etc.
- SampleRate: La frecuencia de muestreo del audio 8000, 44100, etc.
- ByteRate: Es la tasa de datos procesados por unidad de tiempo.
- BlockAling: Es el número de Bytes para una muestra que incluye todos los canales.
- BitsPerSample: Determina cuantos bits habrán en una muestra. 8bits=8, 16bits=16 etc.

Los subchunk “datos” contiene el tamaño de los datos y el sonido real:

- Subchunk2ID: Contiene las cartas de “datos”.
- Subchunk2Size: Este es el número de bytes en los datos.
- Datos: de este punto hasta el final del archivo se ubican los datos de sonido reales.

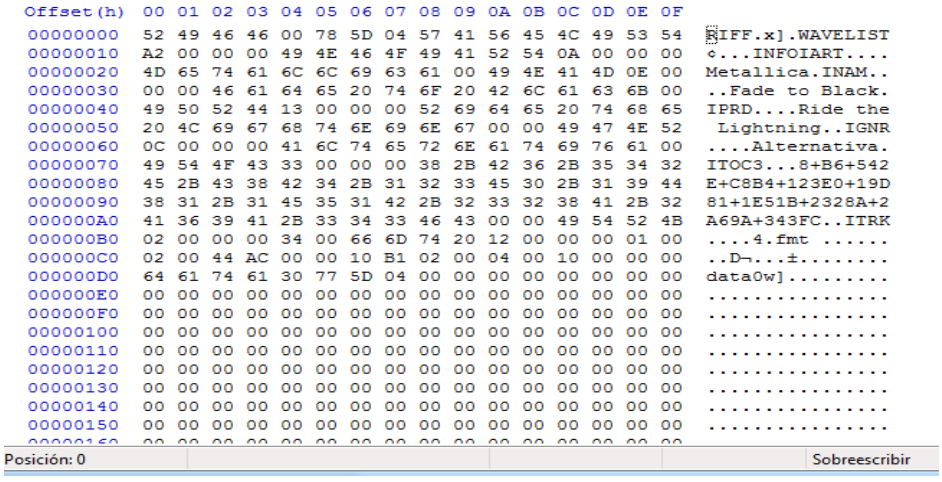


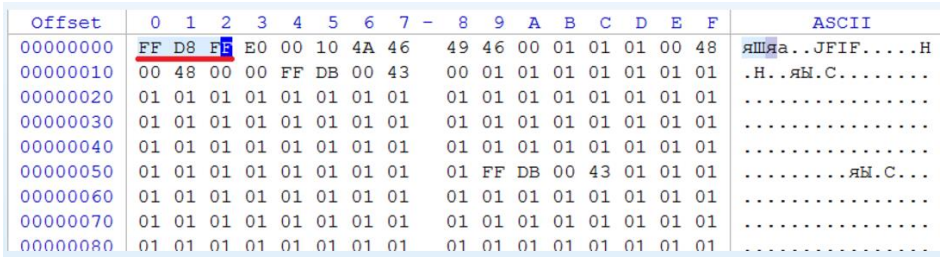
Imagen 4. Muestra de el *header* de un archivo de audio digital WAV

2.4 FORMATO DE ARCHIVO DE IMAGEN JPG O JPEG

JPEG (Joint Photographic Experts Group) es un tipo de formato de imagen con un grado de compresión con pérdidas para imágenes digitales, el grado de la compresión puede ser ajustado para encontrar el balance entre el espacio requerido para almacenamiento y la

calidad de la imagen. Entre los distintos formatos de archivo de imagen JPEG se encuentran JPEG / Exif que es el formato de imagen mas utilizado por las cámaras digitales y otros dispositivos de captura de imagen. JPEG / JFIF es el formato más común para almacenar y transmitir fotografías por internet.

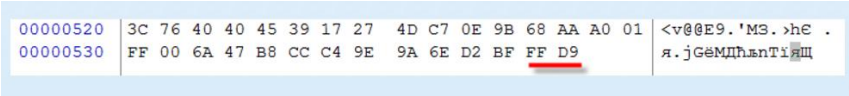
Al analizar la formación de un archivo de imagen JPEG comienzan con un marcador de imagen que siempre contiene los valores hexadecimales del código marcador FF D8 FF. No tiene una longitud del archivo incrustado, por lo que tenemos que encontrar remolque JPEG, que es FF D9.



Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ASCII
00000000	FF	D8	FF	E0	00	10	4A	46	49	46	00	01	01	01	00	48	яШяа..JFIF.....H
00000010	00	48	00	00	FF	DB	00	43	00	01	01	01	01	01	01	01	.H..яЫ.C.....
00000020	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	.....
00000030	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	.....
00000040	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	.....
00000050	01	01	01	01	01	01	01	01	01	FF	DB	00	43	01	01	01	.....яЫ.C...
00000060	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	.....
00000070	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	.....
00000080	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	01	.....

Imagen 5. Vista de primeros pixeles de imagen formato JPEG
Imagen obtenida de http://www.file-recovery.com

Para determinar el tamaño del archivo al no estar designado en el encabezado es necesario desplazarse hasta el punto del remolque JPEG o la finalización del archivo que estipulado anteriormente como FF D9 como se muestra en la siguiente imagen.



00000520	3C	76	40	40	45	39	17	27	4D	C7	0E	9B	68	AA	A0	01	<v@@E9.'MЗ.›hЄ .
00000530	FF	00	6A	47	B8	CC	C4	9E	9A	6E	D2	BF	FF	D9			я.jGёМДЋљnТїяЩ

Imagen 6. Vista del final de imagen formato JPEG
Imagen obtenida de http://www.file-recovery.com

Encabezado de archivos JPEG:

```
Typedef struct _JFIFHeader
{
  BYTE SOI [2]; / * 00h Inicio del marcador de imagen * /
  BYTE APP0 [2]; / * 02h Marcador de uso de la aplicación * /
  Longitud BYTE [2]; / * 04h Longitud del campo APP0 * /
  Identificador BYTE [5]; / * 06h "JFIF" (terminado en cero) Id String * /
  Versión BYTE [2]; / * 07h Formato JFIF Revisión * /
  Unidades BYTE; / * 09h Unidades usadas para Resolución * /
  BYTE Xdensity [2]; / * 0Ah Resolución horizontal * /
  BYTE Ydensity [2]; / * 0Ch Resolución vertical * /
  BYTE X / * 0Eh Conteo de píxeles horizontal * /
  BYTE Y; / * 0Fh Recuento de píxeles vertical * /
} JFIFHEAD;
```

- SOI es el inicio del marcador de imagen y siempre contiene los valores de código de marcador FFh D8h.
- APP0 es el marcador de aplicación y siempre contiene los valores de código de marcador FFh E0h.
- Longitud es el tamaño del segmento de marcador JFIF (APP0), incluyendo el tamaño del campo Longitud en sí y cualquier información de miniatura contenida en el segmento APP0. Debido a esto, el valor de la longitud es igual a 16 + 3 * XThumbnail * YThumbnail.
- El identificador contiene los valores 4Ah 46h 49h 46h 00h (JFIF) y se utiliza para identificar el flujo de código conforme a la especificación JFIF.
- Versión identifica la versión de la especificación JFIF, con el primer byte que contiene el número de revisión principal y el segundo byte que contiene el número de revisión secundaria. Para la versión 1.02, los valores del campo Versión son 01h 02h; Los archivos más antiguos contienen 01h 00h o 01h 01h.

Esta data inicial varia solamente en el tipo del sub formato JPEG con el que se describe. [15]

2.5 BIT

Bit es la abreviación de Binary Digit (digito binario), la cual en términos técnicos es la menor unidad de información de una computadora. Un bit tiene solamente un valor (que puede ser 0ó1).
Varios bits combinados entre sí dan origen a otras unidades, como byte, Mega, Giga y Tera. Toda la información procesada por una computadora es medida y codificada en bits. El tamaño de los archivos es medido en bits, las tasas de transferencia son medidas en bit, toda la información en el lenguaje del usuario es convertida a bits para que la computadora la "entienda", etc.
Los Bits también son utilizados para la clasificación de colores de una imagen. Por ejemplo: una imagen monocromática tiene 1 bit en cada punto (blanco o negro), mientras una imagen de 8 bits soporta hasta 256 colores. [20]

2.6 MSB y LSB (Bits de mayor y menor importancia)

Al momento de realizar la conversión de audio del mundo análogo al mundo digital es importante tener en cuenta la cantidad de bits que se representaría como el rango dinámico digital que tenemos al momento de convertir el audio. Cuando se agrega un bit a un sistema de digitalización de audio, estamos doblando la posibilidad de plasmar en el mundo digital el voltaje del audio análogo. En los sistemas de 16 bits existiría un rango dinámico de -96 DBFS, entiendo el rango dinámico como la diferencia en decibeles que puede tener una señal antes de distorsionar o de enmascararse con el ruido propio del sistema electroacústico.

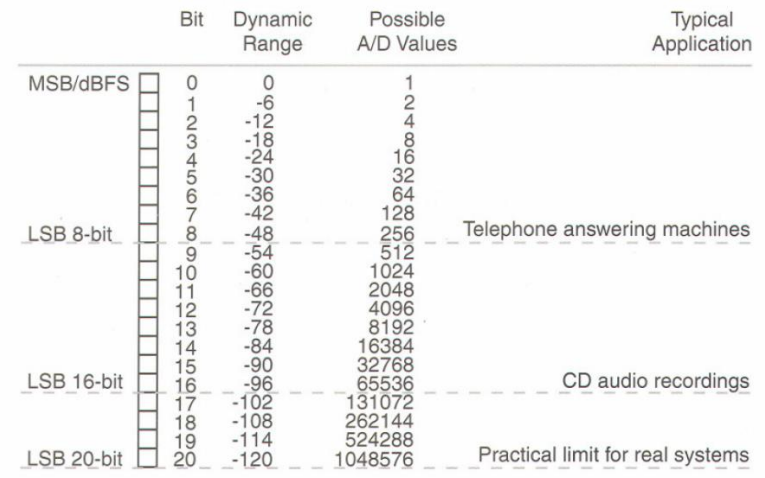


Imagen 7. Ilustración entre profundidad en bits rango dinámico y aplicaciones
imagen tomada de http://www.prosoundtraining.com

En la imagen se describe de mejor manera la relación que existe entre la profundidad en bits para la digitalización del audio y el rango dinámico análogo, en palabras mucho más simples la profundidad en bits del audio digital es el número de posibilidades de la conversión análoga/digital de relacionar un voltaje análogo con una valor digital de cada muestra digitalizando de mejor manera el audio. [7]

Los bit menos significativos son aquellos más alejados del 0DBFS, entre más alto sea el voltaje análogo el conversor análogo / digital generara una muestra con un valor más cercano a los 0 DBFS como se muestra en la imagen de la *imagen 7.*

CAPÍTULO III: DISEÑO METODOLOGICO

3.1 LINEA DE INVESTIGACIÓN DE LA UNIVERSIDAD SAN BUENAVENTURA

El presente proyecto pertenece a la línea de investigación de "DISEÑO DE SISTEMAS DE SONIDO" enfocándola en este caso hacia la programación.

3.2 VARIABLES

- PROFUNDIDAD EN BITS.

"La PROFUNDIDAD DE BITS es determinada por la cantidad de bits utilizados para definir cada píxel. Cuanto mayor sea la profundidad de bits, tanto mayor será la cantidad de tonos (escala de grises o color) que puedan ser representados. Las imágenes digitales se pueden producir en blanco y negro (en forma bitonal), a escala de grises o a color." [7]

- TAMAÑO DE ARCHIVO.

Es la cantidad de espacio en memoria que ocupa un archivo dependiendo del grado de compresión del archivo que no permitirá en algunos casos transferir el archivo ya que tienen una restricción del tamaño a la hora de enviar.

3.3 ENFOQUE DE INVESTIGACIÓN

Para determinar los resultados del proyecto el método cuantitativo es el más favorable debido a que todas las variables que se presentan para su desarrollo pueden ser medidas de forma numérica determinando porcentajes y cantidades exactas en los resultados del desarrollo.

CAPITULO IV: DESARROLLO INGENIERIL

A continuación se presentara el desarrollo del proyecto de grado que lleva como título "Algoritmo para la encriptación y desencriptación entre archivos digitales de audio e imagen" que consta de los siguientes tres grandes aspectos para su desarrollo y puesta en marcha.

DISEÑO DEL ALGORITMO	IMPLEMENTACIÓN DEL ALGORITMO	PRUEBAS

Que se irán llevando a cabo en el orden propuesto anteriormente de forma organizada y consecuente con el fin de cumplir los objetivos anteriormente propuestos y obtener conclusiones y resultados acordes con los lineamientos iniciales del proyecto que es saber si es posible proteger la información por medio de la encriptación pero perdiendo un mínimo de la información.

Otros aspectos relevantes es el manejo de archivos encriptados y su portabilidad en diferentes plataformas actuales como los dispositivos móviles y plataformas de almacenamiento en la nube.

Para el final realizar una comparación con respecto a los antecedentes planteados por el autor Rahul R Upadhyay que es la antesala al proyecto realizado para soportar las conclusiones que del presente desarrollo surjan, determinando un punto de partida para futuras investigaciones y la implementación en diferentes plataformas de procesos de encriptación para más formatos de multimedia digital.

4.1 DISEÑO DE ALGORITMO PARA ENCRIPTACIÓN Y DESENCRIPTACIÓN

Para el diseño y desarrollo del algoritmo de encriptación entre formato de audio WAV y formato de imagen JPEG es importante tener en cuenta los siguientes aspectos:

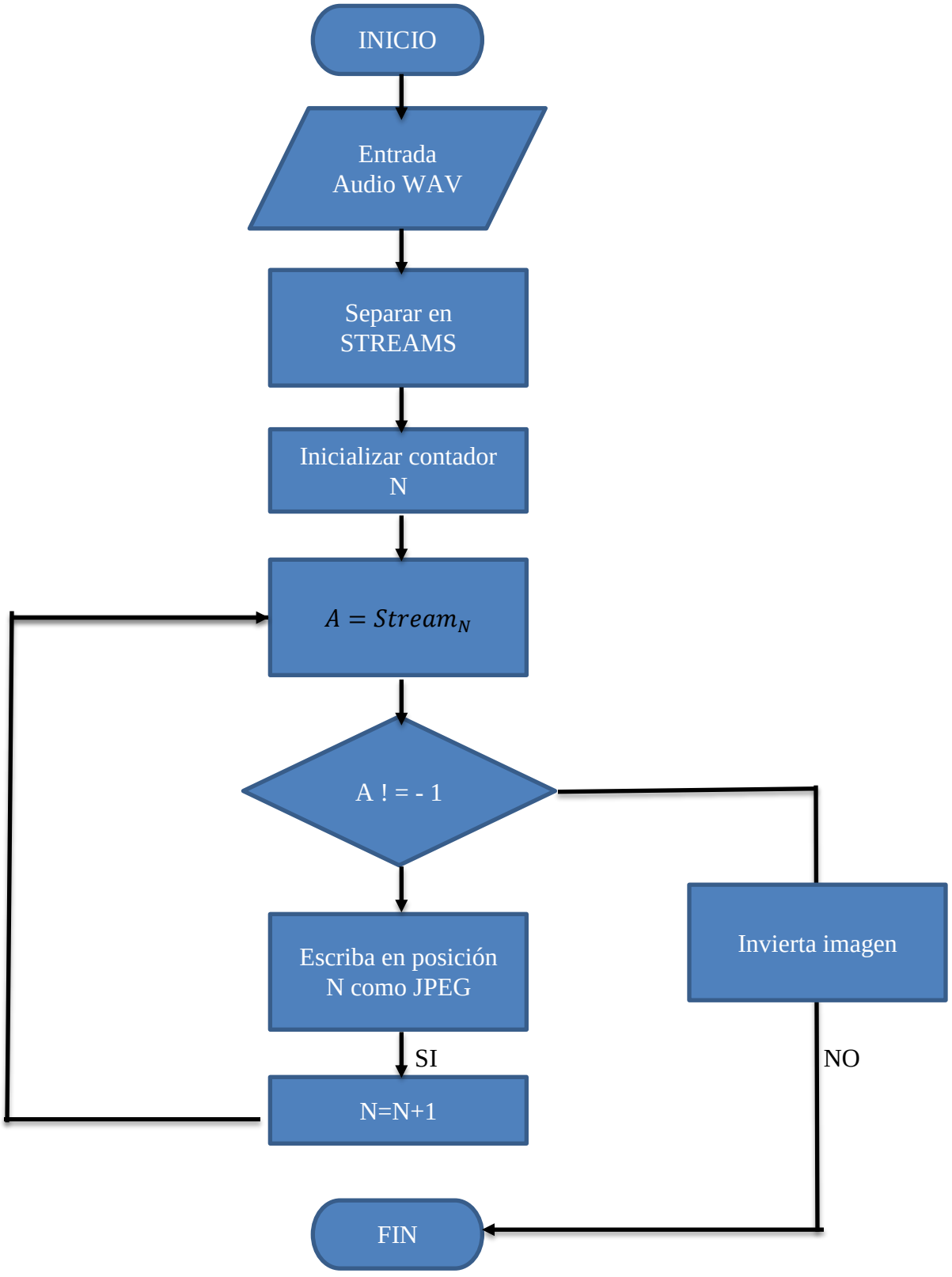
- El cifrado o encriptación simétrica será la mejor opción en cuanto a tiempo de ejecución debido a la cantidad de datos que tiene un archivo de formato WAV.
- La cantidad de datos no debe ser importante a la hora de ejecutar la encriptación.
- Los datos del *header* del audio WAV deben permanecer intactos para no perder las características que significan a la hora de la reproducción *(capitulo 2 apartado 2.3).* Ya que afectar esta información significaría dejar sin formato los datos siguientes del audio WAV.

- Las imágenes JPEG están compuestas por tres matrices de igual tamaño cada una de ellas correspondiente a un color ROJO VERDE AZUL y en cada una tiene 8 bits para llenar cada una de las posiciones, la combinación de los valores de las tres matrices en una posición forman los pixeles de imagen.
- Alterar la información que describe el formato de imagen o la ausencia de ella generaría un error en la lectura en cualquier tipo de visualizador.

Teniendo en cuenta las anteriores consideraciones el algoritmo a diseñar deberá tomar un audio en formato WAV y convertirlo en una imagen JPEG, para ello el punto de relación entre los formatos es el byte *(capitulo 2 apartado 2.5).* Y esta será una de las claves de la encriptación facilitando el proceso y al mismo tiempo aumentando la seguridad y disminuyendo al mismo tiempo la posibilidad de perder información en el momento de la escritura del formato al momento de encriptar.

En el siguiente diagrama se muestra el algoritmo para el cifrado de un archivo de audio WAV en una imagen de formato JPEG.

DIAGRAMA DE FLUJO PARA ENCRIPTACIÓN



EXPLICACION PASO A PASO DEL ALGORITMO

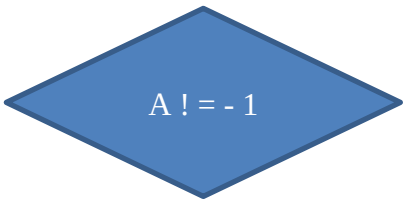
Este recuadro marca el inicio del algoritmo de encriptación.

Al momento de la lectura del audio WAV es importante tener en cuenta que los datos característicos como la frecuencia de muestreo la cantidad de canales y la profundidad en bits, que se encuentra implícita en el *header* y es relevante a la hora de encriptar el audio y que esta además no siempre ocupa el mismo espacio en memoria.

El separar la información del audio en su mínima expresión, que es el lenguaje binario, facilitara en gran manera la escritura del formato de imagen al momento de la encriptación y es aquí donde está la clave para no perder información en el proceso, en este proceso se agruparan los bits en cantidades según lo indica la información de *header* 16 bits 24 bits según sea el audio que ingreso.

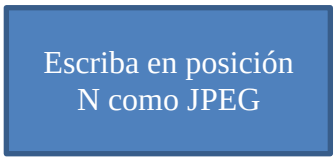
La variable que se inicia en este proceso servirá como contador para iniciar la escritura del formato de imagen y esta ira aumentando conforme se vayan reescribiendo muestras de un formato a otro.

El valor de la variable A ira cambiando conforme N aumente, como consecuencia cada vez que N aumente de uno en uno la variable A se sobrescribirá con uno de los grupos de valores creados anteriormente.

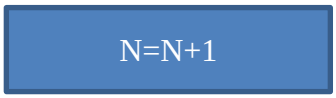


Esta condición analiza el valor de cada uno de los bytes y lo compara con -1 teniendo solo dos posibles respuesta:

- SI



Este proceso escribirá el Stream que indica N en la misma posición N pero ahora siendo parte del formato de imagen JPEG.



Al momento de aumentar la variable N de uno en uno la encriptación no se volverá un ciclo infinito escribiendo siempre el mismo grupo de bits en la misma posición.

- NO

El proceso de la inversión es el paso final y quizás el mas importante a la hora de brindar una protección a la información, ya que de no realizarse el audio puede ser legible por cualquier reproductor ya que los datos del *header* le permitirían hacer lectura y no podría llamarse encriptación. Este es el proceso que definirá la seguridad de la información y puede ser variable. Para este caso aplicaremos la inversión que se mostrara a continuación.

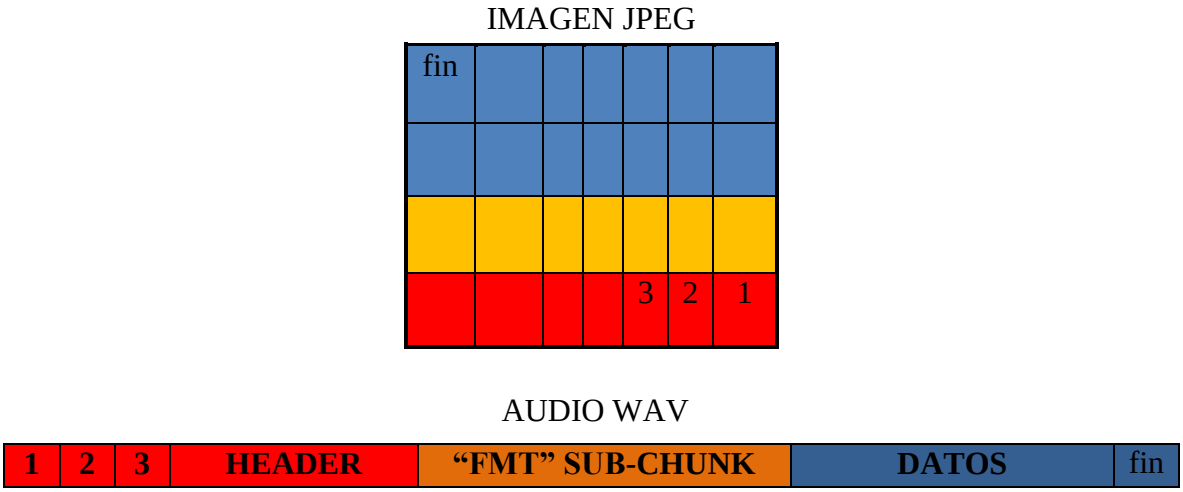


Imagen 8. Ilustración de encriptación audio WAV imagen JPEG

Se alterara el orden de los datos que ya son pixeles de la imagen colocándolos como se observa en la anterior imagen el que antes era el primero pasara a ser el último, el que era el segundo será el antepenúltimo y así sucesivamente.

Al momento de finalizar el algoritmo el resultado será una imagen en formato JPEG con el mismo espacio en memoria que el audio original, y aunque tiene formato de imagen ningún visualizador de archivos de imagen podrá proyectarla y ningún reproductor de archivos de audio será compatible para su reproducción debido a los bytes necesarios para describir el formato o el *header* no están al principio del archivo.

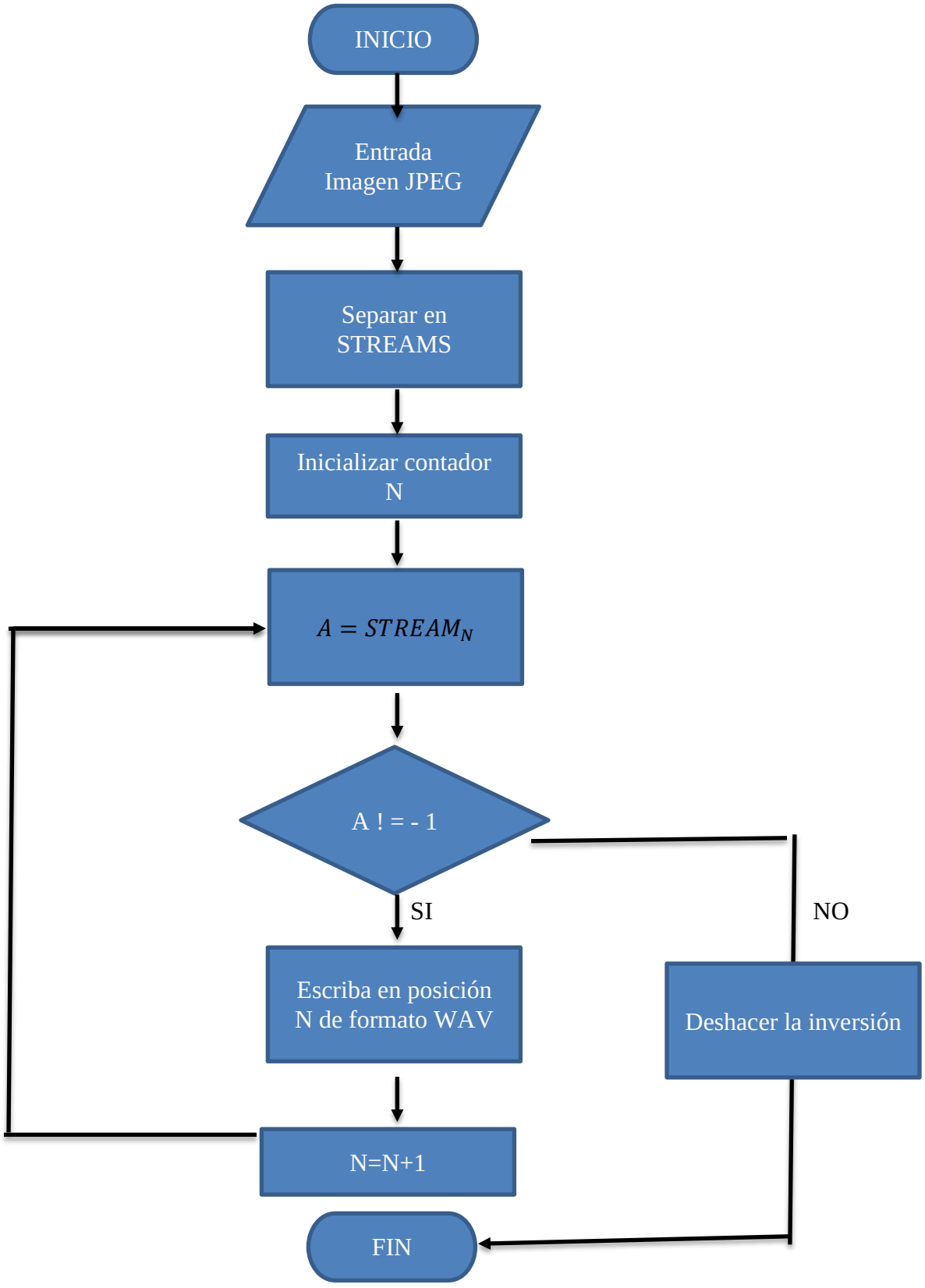
EJECUCIÓN

Al momento de ejecutar el algoritmo el último valor de un audio es un -1 indicando que no existen más muestras después de esta, el algoritmo funciona haciendo una transcripción byte a byte del audio digital ahora escribiéndolo como los pixeles de un formato de imagen teniendo en cuenta que cualquier proceso del mundo digital parte del lenguaje binario, como resultado se obtendrá una imagen JPEG sin sus datos de inicio que describen que formato es el que se está trabajando y logrando así que el archivo sea ilegible que no tenga perdida alguna pero que pueda ser transportado, técnicamente se está omitiendo la escritura de los datos del encabezado o *header* de la imagen JPEG. En esta encriptación se modifica levemente el orden de los datos obteniendo los resultados que se requieren como son:

- Un archivo ilegible.
- Un archivo sin perdidas debido a la encriptación o a la codificación del formato utilizado
- Un archivo portable
- La única forma de restaurar el archivo se la revertir la encriptación.

Aunque de igual manera al momento de diseñar una inversión de los pixeles de imagen al momento de la encriptación entre mas compleja sea compleja aumentara también la seguridad para la información.

DIAGRAMA DE FLUJO PARA DESENCRIPTACIÓN



Para el algoritmo de la desencriptación simplemente se necesita revertir la inversión que se realizó en la encriptación y reutilizar el código para volver a tener el audio WAV completamente legible en la frecuencia de muestreo y la profundidad en bits *(bit depth)* originales. De no deshacer la inversión de los datos realizada en la encriptación el resultado de su contraparte será un archivo en formato de audio WAV que es ilegible por cualquier reproductor.

4.2 IMPLEMENTACIÓN DE LOS ALGORITMOS

La elección del entorno de programación para desarrollar este proyecto es java ya que hay muchas aplicaciones y sitios web que cuya programación se realiza en java y cada día se crean más. Java es rápido, seguro y fiable. Desde portátiles hasta centros de datos, desde teléfonos móviles hasta Internet, lo que le dará al proyecto muchas más posibilidades de poderse implementar en cualquier dispositivo logrando encriptar y desencriptar de manera rápida y en cualquier momento.
Otras de las muchas ventajas que tiene java es el hecho de ser gratuito lo que implica que no es necesario gastar grandes cantidades de dinero en un software que sirva para desarrollar este proyecto, el hecho de que tenga librerías descargables y que puedan desarrollarse facilitan el proceso a la hora de programar además de ser multiplataforma facilitando la masificación de todos los desarrollos en este entorno.
Una de las funciones de java es poder recibir archivos con flujo o STREAMS que es precisamente lo que se necesita para el desarrollo del proyecto es una librería de java que permite trabajar con los archivos dato a dato en audio serian cada una de las muestras que lo componen, y en imagen seria cada una de pixeles, se tomaran archivos de audio WAV e imagen JPEG desde cualquier ruta del ordenador para poder desencriptarla o encriptarla según sea el caso y posteriormente exportarla.
De la misma forma es posible reproducir los archivos que se trabajen sean los encriptados o los desencriptados. La aplicación en java es la siguiente:

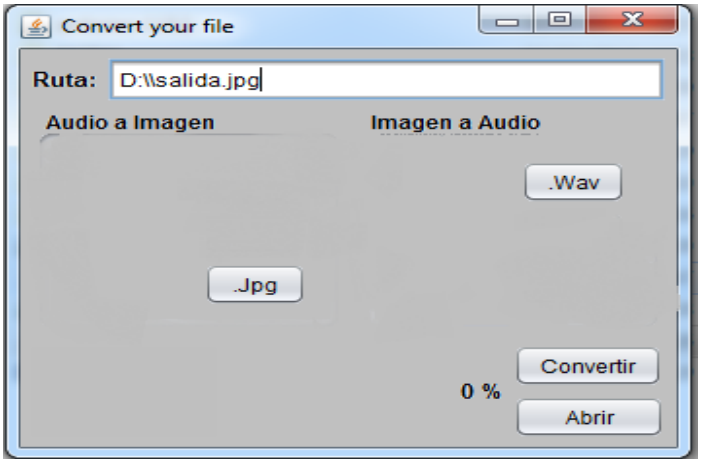


Imagen 9. Interfaz gráfica principal

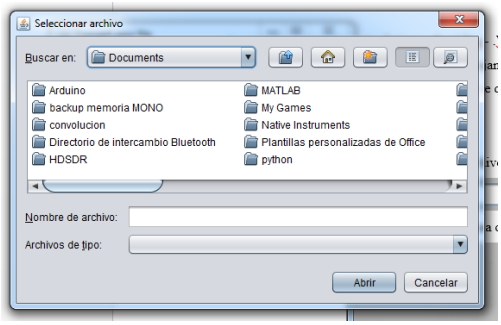
La interfaz consta de cuatro botones dos (.Jpg - .Wav) para seleccionar el formato de salida uno para abrir los archivos que se están trabajando bien sea encriptando o desencriptando (Abrir) y otro para seleccionar el archivo que se quiera ingresar al algoritmo (Convertir).

Pasos

- Determinar la ruta y el nombre del archivo de salida.

- Determinar el formato de salida bien sea con los botones o escribiéndolo en la ruta

- Hacer click en el botón convertir para desplegar la siguiente ventana.



- En esta ventana se selecciona el archivo que se quiera trabajar bien sea encriptando o desencriptando, al momento de hacer click en el botón abrir comienza a ejecutarse el algoritmo.
- Al terminar el proceso en la interfaz principal preguntara si desea abrir o no el archivo que se estuvo trabajando.

RESULTADOS DE LA ENCRIPTACIÓN

Como fue previsto en el diseño el resultado de la encriptación es una imagen en formato JPEG que no puede ser visualizada pero ocupa un espacio en memoria igual al el audio que se encripto inicialmente asegurando que no hay perdidas al momento de encriptar y puede

ser almacenada en diferentes dispsitivos portables como memorias USB dispositivos SmartPhone Dropbox etc.
Las siguientes imágenes mostraran los resultados del proceso de encriptación descrito en el capítulo anterior comparando el archivo que ingresa al algoritmo y el archivo resultante de la encriptación.

Imagen 10. Características del audio que será encriptado.

El archivo que ingreso para ser encriptado tiene una duración de 00:06:55 un tamaño de 69,8MB tiene una frecuencia de muestreo de 44100 Hz y una profundidad de 16 bits (es sacado del CD en físico original).
El *header* de los datos del archivo original antes de ser encriptados de la *imagen 11* muestra datos adicionales como el género, álbum, título y numero de canción dentro del álbum. El resultado de la encriptación es el siguiente.

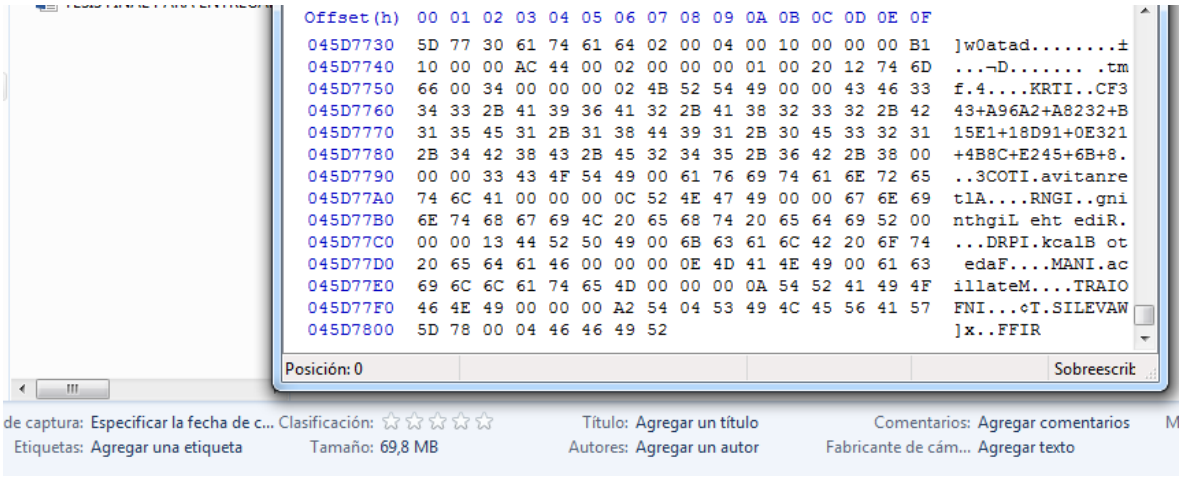


Imagen 11. *header* encriptado aplicando la inversión propuesta en el capítulo 4.1.

Como se observa en la anterior imagen el *header* encriptado está invertido según como propone el capítulo anterior haciendo ilegible el archivo de audio que además de estar invertido por completo carece de la información inicial que usan los reproductores de audio a la hora de identificar un archivo y reproducirlo correctamente. Además se identifica que el resultado de la encriptación tiene el mismo tamaño que el archivo original asegurando que no hay pérdida alguna al momento de encriptar.

4.3 PROTOCOLO DE PUEBAS

Para demostrar que no hay ninguna pérdida que pueda afectar el mensaje que contiene el archivo y que al momento de ser encriptado es completamente portable, se tomara un trozo de audio WAV que tenga un tamaño de archivo menor a 20 MB será encriptado y posteriormente será almacenado en una memoria USB en un SmartPhone será enviado por chat y por WhatsApp para posteriormente ser desencriptado y realizar una correlación con ayuda de Matlab para determinar la perdida de información que pueda tener el archivo después de este proceso y en caso de presentarse alguna perdida examinar en qué punto se pudo haber perdido la información.

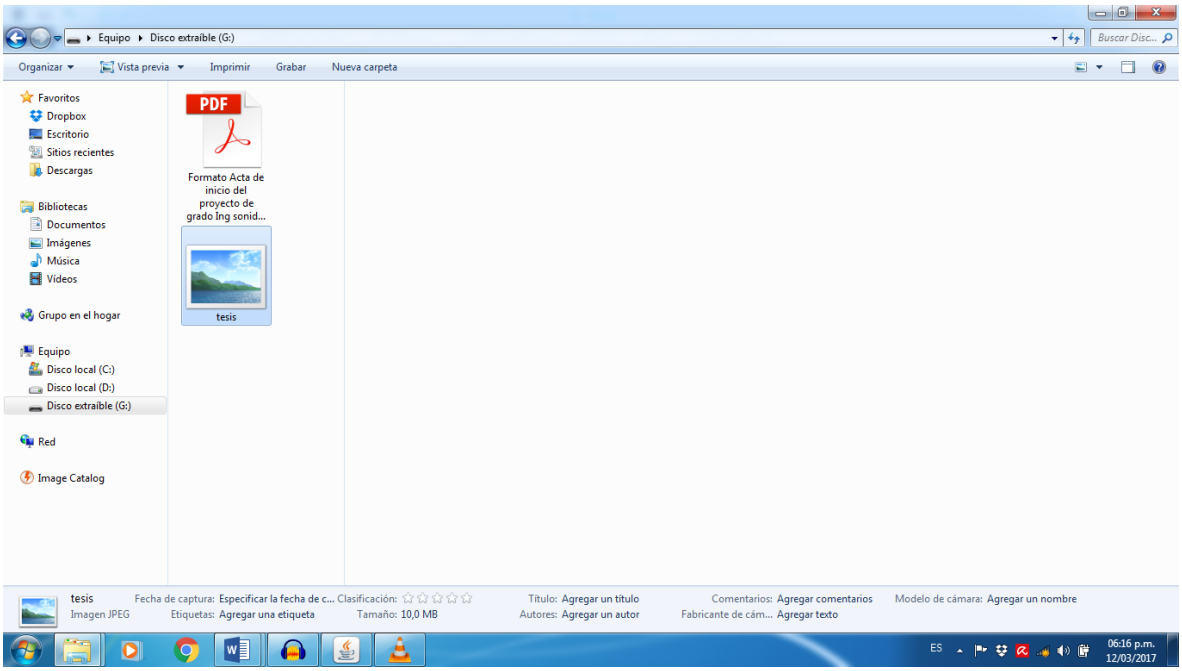


Imagen 12. Archivo encriptado almacenado en una memoria USB

El archivo encriptado tiene un tamaño de 10 MB es un trozo de un minuto del audio trabajado al en el capítulo *4.2* y como se muestra en la anterior imagen ya está en una memoria USB y hasta el momento conserva su tamaño de archivo y la encriptación no permite visualizarlo.

El paso siguiente será todo lo que tenga que ver con dispositivos móviles debido a que actualmente allí es donde se almacena la información personal y se puede visualizar en cualquier momento.

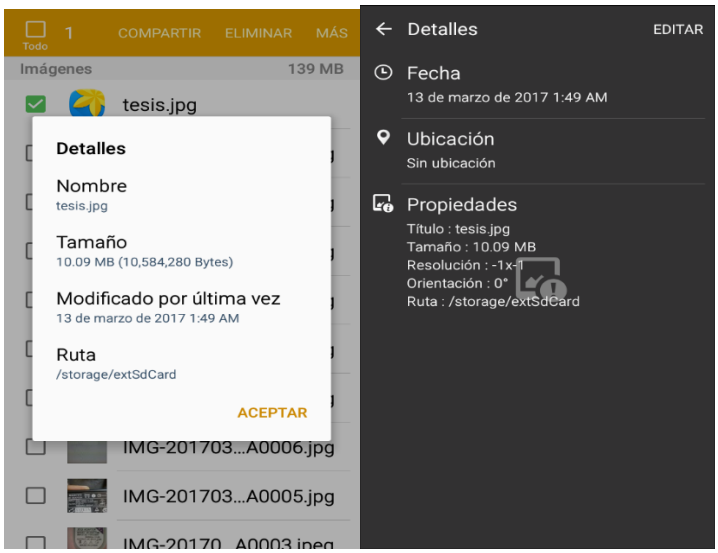


Imagen 13. Imagen visualizada en dispositivo Android

Como se puede observar en la imagen el archivo tampoco puede ser visualizado y tiene el mismo peso que el archivo original y sigue sin poder visualizarse antes ya que aún sigue encriptado.

Al intentar enviar el archivo por WhatsApp ya sea directamente desde el celular o en su servidor web se genera un error debido a que la plataforma no identifica el archivo encriptado como una imagen generando el siguiente error. *(Capitulo 2 item 2.4 estructura Jfif).*

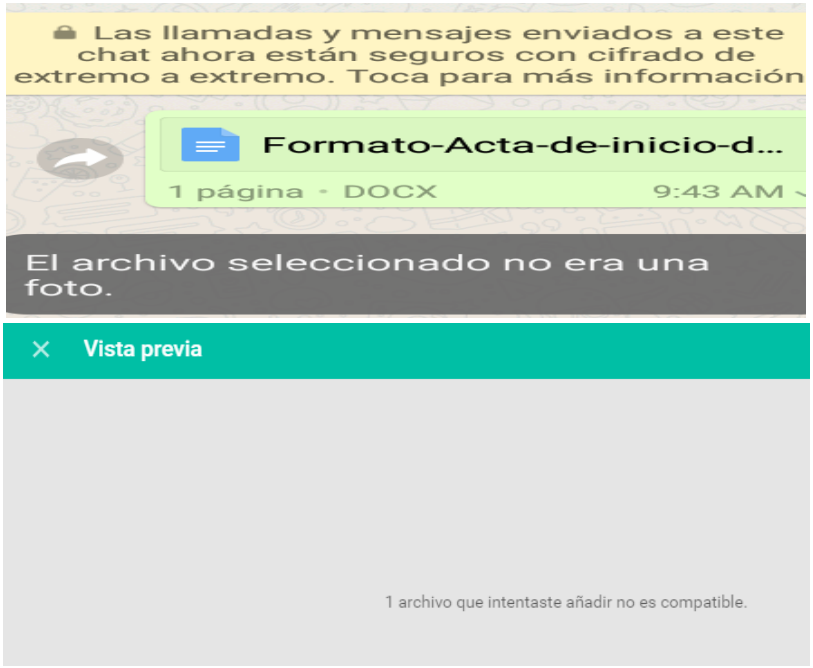


Imagen 14. Error presentado al transfeir la imagen encriptada por WhatsApp

Este error además no puede ser corregido colocando simplemente estos datos como se referencio anteriormente debido a que además de ello el archivo que resulta de la encriptación no es una imagen de 8 bits, por el contrario es un archivo de 16 bits, en otras palabras para hacer que el archivo encriptado fuera apto para ser enviado por este tipo de medios existen diferentes limitantes una de ellas es el tamaño del archivo, debido a que la cantidad de información que puede tener un archivo multimedia es muy pequeña. Además de que en esencia el archivo encriptado es un audio y el objetivo es preservar intacta la información del mismo.
Algunas de las formas de solucionar estos inconvenientes serian tomando el tamaño total del archivo de audio original, dividirlo en Streams y a partir de ahí formar mas de un archivo de imagen para dividir la información adicionando también el *header* necesario para que esta información sea reconocida como imagen y no presente problemas a la hora de ser enviada. Esto cambia como tal el archivo de audio ya los Streams tendrían que ser de 8 bits dividendo cada una de las muestras de audio afectándolo directamente esto salvo que el audio fuera desde un principio de 8 bits.

Al momento de desencriptar el archivo para hacer una correlación entre el audio original y el audio desencriptado será usado el archivo que paso por todas las pruebas para asegurar que no hubo pérdida alguna mientras estas se llevaban a cabo.

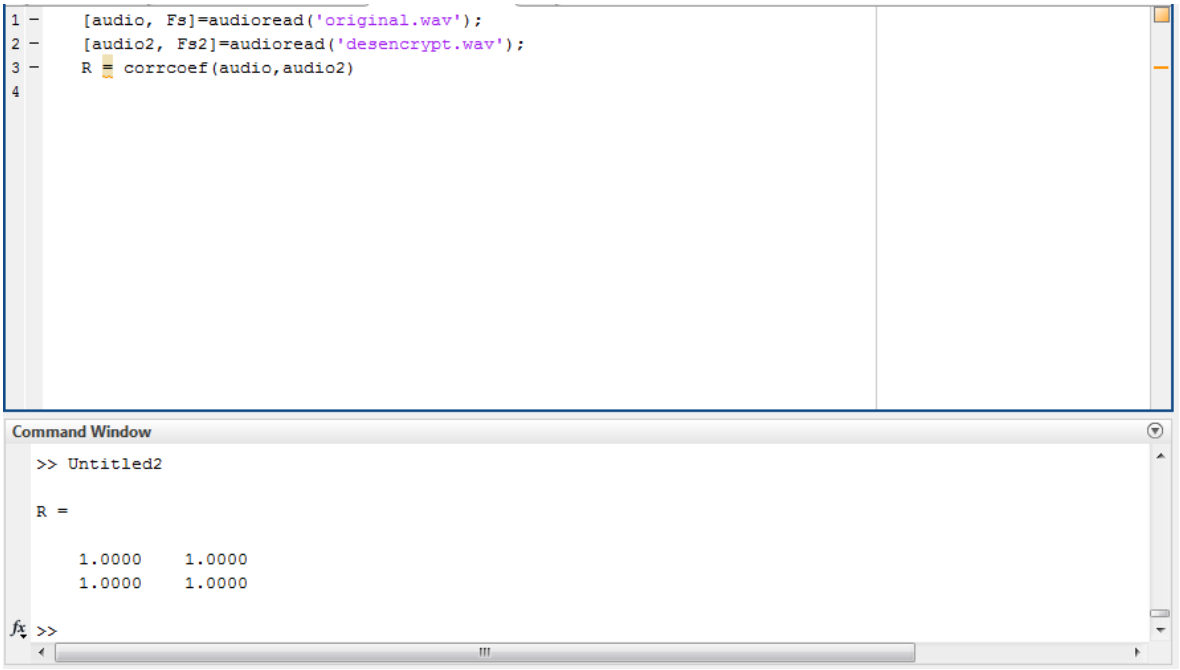


Imagen 15. Coeficiente de correlación entre audio original y audio desencriptado.

La función que Matlab ofrece *corrcoef* permite saber que tan parecida es una señal de la otra esta arroja un resultado cuyo valor mas alto es 1 lo que quiere decir que la señal es la misma y 0 que quiere decir que las señales no son para nada parecidas entre sí.

Como se puede observar en la anterior imagen las señales son exactamente iguales así que no existe perdida en la encriptación aplicando el algoritmo.

4.4 DISCUSIÓN

El trabajo desarrollado por Rahul R Upadhyay en su publicacion "Study of Encryption and Decryption of Wave File in Image Formats" es muy claro a la hora de expresar que el formato JPEG no es útil a la hora de encriptar archivos de audio debido a que tiene una perdida por la compresión que el formato realiza que puede llegar a afectar de forma muy abrupta la calidad del audio, además defiende el uso de formatos de imagen sin compresión como el TIFF para desarrollar este proceso.

Al momento de desarrollar este tipo de proyectos en entornos de programación con una licencia paga como Matlab limita el hecho de poder continuar con la investigación o refutarla debido a que el acceso a ellos es muy restringido, otra de las desventajas grandes que tiene Matlab es que las librerías que implementa a la hora de trabajar con audio no respetan en lo absoluto el proceso de digitalización del mismo debido a que los valores de las muestras que lee están normalizados entre 1 a -1 y no permite saber aspectos vitales a la hora de trabajar con audio digital como la frecuencia de muestreo y la profundidad en bits con la que viene el audio que ingresa a Matlab.

Otro de los factores y quizás el mas importante es que en ningún punto del desarrollo de su investigación el autor explica cómo se forma el audio en formato WAV debido a que Matlab omite esta información inicial *header* a la hora de leer el audio, tampoco como se forma el formato de imagen TIFF. Uno de los problemas con el uso de este formato es que es poco popular, dispositivos electrónicos como SmartPhones Tablets y de mas no permiten visualizar estos archivos sin antes descargar una aplicación o un complemento especializado que brinde esta posibilidad, ventaja que si tiene el formato JPEG que puede ser visualizado en muchos más dispositivos sin necesidad de alguna extensión de formatos.

Una de las formas de encriptación que en el documento se plantean requiere pasar las muestras de 16 a 8 bits eliminando los LSB (*capitulo 2 ítem 2.6)* afectando de por si la calidad del archivo de audio sin pasarlo aun al formato JPEG.

Otra forma de lograr la encriptación entre formatos WAV y formatos de imagen como los JPEG sin tener pérdidas es la que se plantea en el presente proyecto de grado que consiste a grandes rasgos en dividir cada sample de audio en su parte mas pequeña que son los bits hacer buen uso del *header* del formato WAV para agrupar estos bits en la cantidad que indica el formato creando bytes, y encriptar no alterando los bytes sino alterando el orden de los mismos obteniendo una encriptación sin perdidas en un formato de imagen mucho mas utilizado como el JPEG y que se pueda desarrollar en cualquier entorno de programación ya que todos brindan la posibilidad de expresar en números binarios sus variables o sus entradas desde disco.

CONCLUSIONES

- El algoritmo de encriptación toma como entrada un archivo audio en formato WAV con cualquier frecuencia de muestreo, profundidad de bits, nombre y tamaño; logra encriptarlo en una imagen de formato JPEG que es ilegible a menos que se aplique la desencriptación para obtener nuevamente el audio formato WAV.
- La implementación del algoritmo en el lenguaje JAVA permite que este pueda ser aplicado en diversas plataformas. El resultado de ello es una encriptación completamente funcional, que aumentara su nivel de seguridad al momento de modificar el orden de la información encriptada.
- Al archivo resultante de la encriptación, aunque no presenta pérdida alguna, no es del todo portable, debido a que para ser transferido en medios como WhatsApp sería necesario agregar información de *header* que permita identificar los Streams como una imagen compatible, esto no solo aumentara el tamaño del archivo sino que deberá ser eliminada en la desencriptación, ya que se afecta la reproducción del archivo recuperado al revertir el proceso de cifrado.
- El algoritmo diseñado aplica la forma de encriptación más clásica pero adiciona un plus que es el manejo mucho más avanzado de los formatos de fuente y de destino y es en este punto donde se logra no solo evitar en absoluto las perdidas sino hacer mucho más difícil reconstruir el archivo de audio WAV sin tener la llave indicada, esto debido a que las combinaciones posibles con las muestras tomadas en un solo segundo más los datos del *header* son demasiadas y adicionalmente no cualquier programa permitirá trabajar con el archivo encriptado.

RECOMENDACIONES

A continuación se plantearan las recomendaciones:

- Aplicar el algoritmo de encriptación entre distintos formatos de audio .MP3 .AIFF y de imagen PNG TIFF entre otros.
- Implementar este tipo de encriptación en plataformas propias de Android y IOS a manera de una aplicación descargable que al momento que reciba un audio este sea encriptado de inmediato
- Trabajar en un formato de compresión de audio que siga siendo lo suficientemente competente para ser usado en audio profesional.
- Trabajar cifrados de tipo asimétrico para lograr el más alto grado de seguridad teniendo en cuenta la gran cantidad de información a encriptar, el manejo de los formatos y la obvia necesidad de no perder información.

REFERENCIAS

[1] J. J. E. Elizondo, FUNDAMENTOS DE PROCESAMIENTO DE IMÁGENES, Mexicali: Universidad Autónoma de Baja California, 2002.

[2] P. ESPECTADOR, «Lanos S.A.,» 20 ABRIL 2015. [En línea]. Available: http://www.espectador.com/tecnologia/313998/conozca-la-primera-cancion-impresa-en-3d.

[3] J. Mora Pascual, H. Mora Mora, A. Fuster Guilló y J. Azorín López, «Adjustable compression method for still JPEG images,» *ELSEVIER,* p. 16, 2015.

[4] B. Tian, S. Sambasivam y J. Barron, «Practical Digital Playback of Gramophone,» *AES,* p. 7, 2011.

[5] M. A. M. Manzano, «PROCESAMIENTO Y ANALISIS DIGITAL DE IMAGENES,» HUAJUAPAN DE LEON, 2009.

[6] O. Niemeyer y B. Edler, «Detection and Extraction of Transients for Audio Coding,» *Audio Engineering Society,* pp. 20-23, 2006.

[7] R. R. Upadhyay, «Study of Encryption and Decryption of Wave File in Image Formats,» *BBD National Institute of Technology and Management,* 2010.

[8] Redacción Periodico El Heraldo, «EL HERALDO,» 16 SEPTIEMBRE 2016. [En línea]. Available: http://www.elheraldo.hn/entretenimiento/1000501-469/divulgan-fotos-%C3%ADntimas-de-la-actriz-estadounidense-jennifer-lawrence.

[9] Redacción Diario El Mundo, «EL Mundo,» 7 Diciembre 2010. [En línea]. Available: http://www.elmundo.es/elmundo/2010/12/07/cultura/1291725335.html.

[10] U. P. d. Madrid, «Matematica aplicada a las tecnologias de la informacion y las comunicaciones,» [En línea]. Available: http://www.dma.fi.upm.es/recursos/aplicaciones/matematica_discreta/web/aritmetica_modular/criptografia.html. [Último acceso: 30 Marzo 2017].

[11] A. Menezes, P. Van Oorschot y S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.

[12] H. Corrales, C. Cilleruelo y A. Cuevas, «Universidad de Alcalá,» 2014. [En línea]. Available: http://www3.uah.es/libretics/concurso2014/files2014/Trabajos/Criptografia%20y%20Metodos%20de%20Cifrado.pdf.

[13] Microsoft, [En línea]. Available: https://msdn.microsoft.com/es-es/library/windows/desktop/ee415713(v=vs.85).aspx.

[14] C. S. Sapp, «Soundfile,» 29 Agosto 2004. [En línea]. Available: http://soundfile.sapp.org/doc/WaveFormat/.

[15] Active File Recovery, «Active File Recovery,» [En línea]. Available: http://www.file-recovery.com/jpg-signature-format.htm.

[16] F. Barletta, M. Pereira, V. Robert y G. Yoguel, «Argentina: dinámica reciente del sector de software y servicios informáticos,» *Revista de la CEPAL,* nº 110, pp. 137-155, 2013.

[17] M. Choy y G. Chang, «Medidas macroprudenciales aplicadas en el Perú,» Banco Central de Reserva del Perú, Lima, 2014.

[18] J. P. García Nieto, Consturye tu Web comercial: de la idea al negocio, Madrid: RA-MA, 2013.

[19] R. Wittmann, «¿Hubo una revolución en la lectura a finales del siglo XVIII?,» de *Historia de la lectura en el mundo occidental*, México D.F., Santillana, 2006, pp. 435-472.

[20] R. Ikeda, «data.path,» Madrid, 2013.

[21] Redacción Publinews Guatemala, «Publinews Guatemala,» 2015 Mayo 2015. [En línea]. Available: https://www.publinews.gt/gt/guatemala/2015/05/25/roban-celular-conductora-diana-guerra-difunden-fotografias-privadas.html.

[22] Java corporation, «Java Oracle,» [En línea]. Available: https://www.java.com/es/download/faq/whatis_java.xml.

ANEXOS

ANEXO 1: CODIGO DEL ALGORITMO EN JAVA

Cabe resaltar que la implementación del algoritmo presentado cambiara en cuanto a sintaxis según sea el lenguaje de programación en el que se desarrolle. Además, en el mismo documento se plantea un tipo de inversión de los Streams que puede ser modificada sin afectar el resultado final.

Teniendo en cuenta lo anterior se presentara el código implementado para hacer la transcripción entre los formatos de audio WAV y de imagen JPEG.

```
1. InputStream is = new FileInputStream(chooser.getSelectedFile());
2. FileOutputStream fos;
3. fos = new FileOutputStream(jTextField1.getText());
4. int a;
5. while ((a = is.read()) != -1) {
6. fos.write(a);
   }
7. Inversión o desinversión
8. fos.close();
```

A continuación serán explicadas una a una las líneas de código presentadas anteriormente:

1. se aplica la función *ImputStream* para asignar a la variable *is* el archivo que ingresara en el formato bien sea para encriptar o desencriptar, el archivo es seleccionado desde la interfaz gráfica e ingresa por medio de la función *chooser.*
2. La función *FileOutputStream* brindara a la variable *fos,* el formato de salida de los Streams que utilizaremos bien sea para escribir la imagen encriptada o para retomar al audio original.
3. La variable *fos* tomara las características de nombre y formato que están escritas en el espacio para texto dentro de la interfaz gráfica.
4. Se inicializa una variable de tipo entero *a.*
5. Se inicia un ciclo con la condición de que *a* tomara uno a uno los Streams de la variable *is* y si este valor es diferente a -1 realice la acción del ítem 6.
6. En la variable *fos* se escribirá el valor de *a.*
7. Luego de terminar el ciclo, se realiza la inversión o el opuesto de la misma según sea el caso.
8. Se cierra la variable *fos.*

Nota: las funciones *read write close* son utilizadas para manejar los Streams y trabajan uno a uno dentro de las variables donde estos se almacenen.