UNIVERSIDAD DE QUINTANA ROO
DIVISIÓN DE CIENCIAS E INGENIERÍA

IMPLEMENTACIÓN DE UN ALGORITMO COMPUTACIONAL PARA ESTEGANOGRAFÍA BASADO EN TÉCNICAS DEL BIT MENOS SIGNIFICATIVO

TRABAJO DE TESIS
PARA OBTENER EL GRADO DE
INGENIERA EN REDES

PRESENTA
ARACELI DE LA CRUZ FRANCO

DIRECTOR DE TESIS
DR. JAVIER VÁZQUEZ CASTILLO

ASESORES
M.T.I VLADIMIR CABAÑAS VICTORIA
DRA. MARIA ISABEL ROCHA GASO
MTI. MELISSA BLANQUETO ESTRADA
MSI. RUBÉN ENRIQUE GONZÁLEZ ELIXAVIDE

CHETUMAL QUINTANA ROO, MÉXICO, AGOSTO DE 2017

UNIVERSIDAD DE QUINTANA ROO
DIVISIÓN DE CIENCIAS E INGENIERÍA

TRABAJO DE TESIS ELABORADO BAJO SUPERVISIÓN DEL COMITÉ DE ASESORÍA Y APROBADO COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE:

INGENIERA EN REDES

COMITÉ DE TRABAJO DE TESIS

DIRECTOR:

DR. JAVIER VÁZQUEZ CASTILLO

ASESOR:

M.T.I VLADIMIR CABAÑAS VICTORIA

ASESORA:

DRA. MARIA ISABEL ROCHA GASO

Universidad de Quintana Roo
DCI DIVISIÓN DE CIENCIAS E INGENIERÍA

CHETUMAL QUINTANA ROO, MÉXICO, AGOSTO DE 2017

Agradecimientos

A Dios por sus infinitas bendiciones por darme fuerza y entrega para concluir esta etapa importante en mi vida.

A mis padres; papá Daniel y mamá Magdalena. De todo corazón les agradezco el haber estado conmigo siempre apoyándome, no solo en lo económico sino también en lo emocional y hacerme una persona de bien a pesar de las adversidades para lograrlo, esta meta que no es únicamente un logro mío, sino también de ustedes. Por demostrarme el esfuerzo incansable, dedicación, constancia y perseverancia que sin duda alguna fueron el motivo de este logro.

A toda mi familia que estuvo presente en especial mi hermano Juan Daniel; que más que un hermano fue mi amigo, mi consejero que siempre me escuchó y dio un sabio consejo para tomar decisiones importantes en mi vida. Y otro ser humano que desde el cielo me cuida, él es mi querido abuelo.

Agradezco a mi comité de tesis por la ayuda proporcionada, en especial al profesor Vladimir Cabañas Victoria y a mi director, el Dr. Javier Vásquez Castillo, que con su experiencia, consejos y responsabilidad ha contribuido a realizar este trabajo. Destacando la disponibilidad para escuchar y resolver cualquier duda que surgió durante el desarrollo.

De igual manera agradezco a todos los profesores universitarios que me transmitieron sus conocimientos y experiencias, loa cuales han sido y serán importantes para mi formación personal y profesional.

Quiero agradecer a mi tutor el profesor Rubén E. González Elixavide por su apoyo incondicional durante mi etapa como Universitaria, su disponibilidad, sus consejos que fueron muy importantes para este logro. Por la confianza de poder concluir la universidad y los conocimientos adquiridos cuando hice mi servicio social en esta Universidad.

Finalmente, un agradecimiento muy especial a todas aquellas personas que de manera directa o indirecta me han brindado su valiosa ayuda para la culminación de este objetivo, me refiero a mis amigos. Aquellos seres que durante esta etapa estuvieron presentes, compartiendo experiencias, alegrías, clases, desvelos, etc. A todos ustedes mil gracias: Jairo, Walter, Aida, Oscar, Luis, Misael, Carlos, Deysi, Mauricio, Bryan, Reyes, Irma, Candy, Laura, Mauricio Morfin.

Dedicatoria

A mi familia principalmente mis padres por su apoyo incondicional brindado durante este proceso, que sin duda alguna no fue fácil no solo para mí, sino, tampoco para ellos. Que, con su esfuerzo juntos hemos logrado esta meta. A ustedes que día a día me demostraron que no existen barreras cuando quieres lograr algo, que las adversidades no son impedimentos, que cada logro es una gran contribución para crecer emocionalmente, laboralmente, pero sobre todo como ser humano. Que con su humildad y entrega formaron de mí una persona con valores, capaz de demostrar a los demás una gran esencia de humildad, con lo que sin duda alguna voy a ejercer esta profesión de todo corazón sin nunca olvidar de dónde vengo y hacia donde quiero llegar.

.

Resumen

El presente trabajo de tesis está basado en la implementación del algoritmo esteganográfico *least significant bit* (LSB, bit menos significativo) para imágenes en escala de grises, con el objetivo de demostrar que es posible ocultar información de forma confiable y eficiente en imágenes sin impactar en pérdidas de datos. La esteganografía puede ser implementada en archivos digitales como imágenes, audio o video.

En el capítulo 1 se presentan los objetivos de este proyecto, señalando el alcance del mismo y la metodología para llevarse a cabo.

En el Capítulo 2 se exponen los conceptos básicos de la criptografía, las características básicas de un sistema de esteganografía, los algoritmos esteganográficos para imágenes que existen, especialmente el algoritmo LSB, por su fácil implementación y rapidez, para finalizar con la relación entre la criptografía y la esteganografía.

En el Capítulo 3 se describe la implementación de el algoritmo LSB en MATLAB, describe el código desarrollado, se realiza un análisis de resultados obtenidos al ocultar información en imágenes digitales usando el algoritmo esteganográfico LSB, para lo cual como primer punto se procede a explicar todas las características de necesarias para la implementación de este algoritmo en imágenes. El algoritmo propuesto, incluye una técnica de aleatorización la cual busca evitar que un atacante descubra el mensaje transmitido de forma sencilla.

En el Capítulo 4 se incluyen los resultados que demuestran que nuestro mensaje puede ser transmitido y recuperado de forma satisfactoria.

Contenido

Índice de Ilustraciones

Índice de tablas

Capítulo 1 Introducción

Introducción general

La esteganografía conocida como el arte de ocultar mensajes sin ser detectados, ha despertado gran interés por militares, o personas civiles con diferentes propósitos. La facilidad de ocultar información sin ser detectada en diferentes medios como imágenes digitales, archivos de audio o videos ha posibilitado que nuevos esquemas de envío de información sean posibles y sin que la seguridad de cierta información confidencial sea altamente comprometida. Este proyecto busca investigar y presentar las distintas técnicas de esteganografía reportadas en la bibliografía abierta, así como también, implementar algoritmos de esteganografía basados en la técnica del bit menos significativo (LSB por sus siglas en inglés). Hoy en día, es común que este tipo de técnicas sean utilizadas en el comercio durante el proceso de firmas digitales, aplicaciones militares, y en general para el ocultamiento de información sensible para un determinado usuario.

Entre las técnicas de esteganografía más utilizadas se encuentra la técnica de sustitución de bits, que es la de sustitución de los bits menos significativos de los cuales se compone un archivo con información. Por ejemplo, cualquier archivo multimedia (archivo con información de audio, imágenes, etc) contiene áreas de datos poco significativos, los cuales se pueden sustituir por otros datos, realizando cambios que son inapreciables visualmente (o auditivamente). Con lo anterior, se permite ocultar información dentro de un archivo portador, haciendo que el mismo parezca igual al original.

Por ejemplo, en el método LSB, 1 bit consiste en alterar el bit menos significativo de las muestras de las que se compone el portador (archivo con información), siendo esta la razón de la denominación de la técnica “Least Significant Bit o LSB”. La teoría nos dice que con la sustitución de los bits menos significativos no se pueden detectar los cambios de una manera sencilla. Una característica, es que el método de sustitución LSB no incrementa el tamaño del archivo portador. También, es posible implementar técnicas LSB de más de un 1 bit; sin embargo, es el archivo con información a enviarse sufrirá un deterioro o distorsión impactando en la calidad de la información a trasmitir (E.d., tratándose de una imagen, ésta podrá cambiar su intensidad en los pixeles cuando se vea alterada). Es por ello que resulta interesante, investigar los distintos niveles de distorsión

que son introducidos en la información a medida que se utilizan más bits para llevar a cabo la técnica de LSB.

Definición del problema:

Durante muchos años hemos sido testigos y víctimas de actos delictivos, donde podemos observar el nivel de peligro que existe, y la inseguridad en el ámbito de comunicaciones son cada vez más constantes y mayores. Por lo tanto, muchos de nosotros nos vemos afectados ante estos actos de delincuencia.

La esteganografía busca ocultar información en diferentes medios de transmisión de datos, el objetivo de esta técnica ha sido guardar en secreto mensajes que solo puedan ser revelados a un grupo de personas que sepan sobre la técnica que está en uso, para lo cual se emplean distintos canales de comunicación se logrando esconder el mensaje de tal manera que no es perceptible. Agregando una técnica para la aleatorización de posiciones a insertar el mensaje, de tal manera que sea impredecible de averiguar en donde se realizó la modificación del bit para insertar el valor correspondiente al mensaje.

1.3 Justificación

Actualmente los esquemas de esteganografía presentados en la literatura abierta tienen un nivel computacional significativamente alto (complejidad computacional). En este sentido, el trabajo buscará presentar un algoritmo computacional para implementar técnicas de esteganografía basada en el LSB por lo que es necesario agregar algunas características que garanticen la seguridad y confiabilidad de la información.

1.4 Objetivos

Objetivo general

Diseñar un algoritmo computacional de esteganografía que implemente la técnica del bit menos significativo.

Objetivos específicos

1. Estudiar los algoritmos para implementar esteganografía basados en la técnica de los bits menos significativos.
2. Medir la degradación introducida en la señal original debido a la distorsión por sustitución de los bits menos significativos.
3. Estudiar las técnicas de aleatorización existentes.
4. Proponer un algoritmo para esteganografía basado en la técnica del bit menos significativo
5. Proponer una estructura de aleatorización que evite conocer la secuencia de remplazo de bits.
6. Pruebas del algoritmo implementado.
7. Redacción del documento final de tesis.

1.5 Alcance

Los alcances están definidos en los objetivos particulares, y se cuenta con bibliografía suficiente del tema en lo que se refiere a algoritmos para esteganografía por lo que los alcances van a ser cumplidos. Así mismo, se comenta que no se tienen limitaciones en cuanto al desarrollo del presente trabajo.

1.6 Metodología

Modelo en Cascada, también llamado Lineal secuencial, es el enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo de software, de tal forma que el inicio de cada etapa debe esperar a la finalización de la etapa anterior.

1.6.1 Fases del modelo en cascada

- **Análisis de requerimientos:** En esta fase se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. Por lo que debemos realizar un documento de especificación de requisitos, que contiene la especificación completa de lo que debe hacer el sistema sin entrar en detalles internos.
- **Diseño del sistema:** Se descompone y organiza el sistema en elementos que puedan elaborarse por separado, aprovechando las ventajas del desarrollo en equipo. Como resultado surge el (Documento de Diseño del Software), que contiene la descripción de la

estructura relacional global del sistema y la especificación de lo que debe hacer cada una de sus partes, así como la manera en que se combinan unas con otras. Es conveniente distinguir entre diseño de alto nivel o arquitectónico y diseño detallado. El primero de ellos tiene como objetivo definir la estructura de la solución (una vez que la fase de análisis ha descrito el problema) identificando grandes módulos (conjuntos de funciones que van a estar asociadas) y sus relaciones. Con ello se define la arquitectura de la solución elegida. El segundo define los algoritmos empleados y la organización del código para comenzar la implementación.

- **Diseño del programa:** En esta fase se realizan los algoritmos necesarios para el cumplimiento de los requerimientos del usuario, así como también los análisis para saber que herramientas usar en la siguiente etapa, la codificación.
- **Codificación:** Es la fase en donde se implementa el código fuente, realizando pruebas y ensayos para corregir errores. Dependiendo del lenguaje de programación y su versión se crean las bibliotecas y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido.
- **Pruebas:** se debe comprobar que el programa funciona correctamente y que cumple con los requisitos, antes de ser entregado al usuario final.
- **Mantenimiento:** Una de las etapas más críticas, ya que se destina un 75% de los recursos, es el mantenimiento del Software, puesto que al utilizarlo como usuario final puede ser que no cumpla con todas las expectativas.

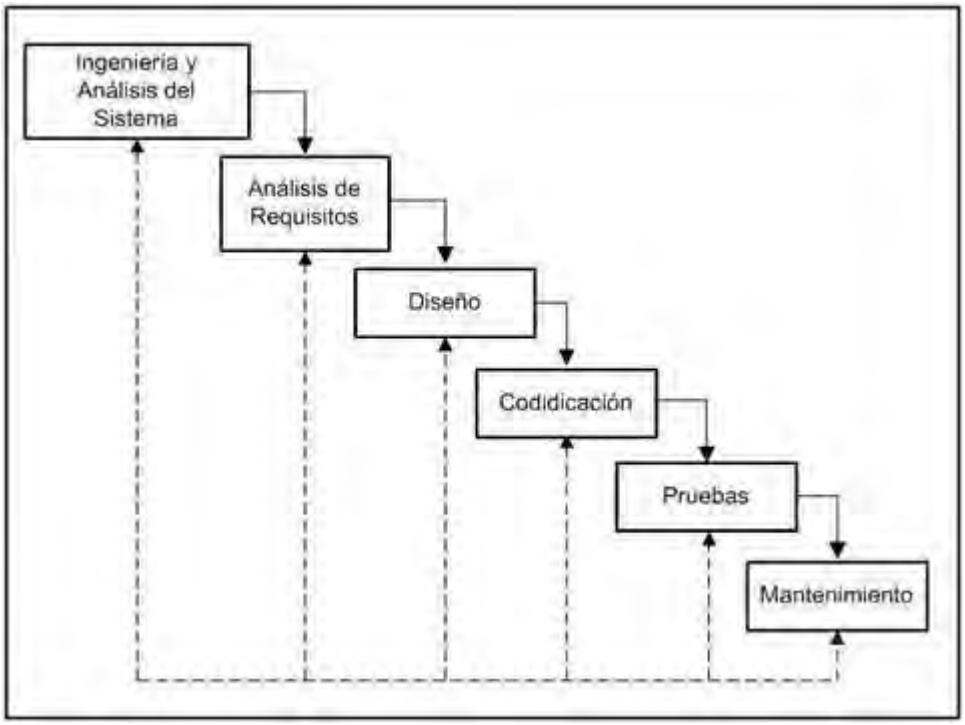


Ilustración 1 Fases del modelo en cascada

Este trabajo de tesis se realizó mediante el modelo de desarrollo conocido como cascada, tomando en cuenta las seis fases que se describieron anteriormente. Sin embargo, cabe mencionar que la fase seis no se está tomando en cuenta para este trabajo porque el diseño del algoritmo no requiere de alguna técnica de mantenimiento.

Capítulo 2 Marco Teórico

2.1 Introducción

La esteganografía es una técnica que permite entregar mensajes ocultos dentro de un archivo digital, de forma que no se detecte su presencia y pasen inadvertidos. Asociada con cifrado de mensajes, la esteganografía puede ser útil para almacenar datos en la nube que queramos tener disponibles desde cualquier lugar. En el campo de la esteganografía digital es posible ocultar información en todo tipo de soportes como archivos de audio, imágenes, vídeos, textos, etc. Una forma muy conocida es la del bit menos significativo (LSB) que pertenece a los llamados métodos de sustitución, consistente en ocultar el mensaje en el bit menos significativo de un archivo, generalmente una imagen, aunque es aplicable a más soportes, de forma que el cambio en el fichero original sea casi imperceptible.

2.2 Seguridad

La necesidad de seguridad de la información ha cambiado en las últimas décadas. Antes del uso de las computadoras, la seguridad de la Información era proporcionada por medios físicos, por ejemplo, el uso de cajas fuertes y por medidas administrativas, como los procedimientos de clasificación de documentos. Con el uso de la computadora, y más aún con la llegada de Internet, fue indispensable el uso de herramientas automatizadas para la protección de archivos y otro tipo de información almacenada en la computadora, algunas de estas herramientas son los cortafuegos, los sistemas detectores de intrusos, el uso de sistemas criptográficos o estenográficos. Estas herramientas permiten proteger a la información, además de los sistemas informáticos que son los encargados de administrar la información. De la necesidad por proteger a la información y a los sistemas que la administran surge el término de Seguridad Informática. Tal como menciona [1] “el hecho de que actualmente los términos de seguridad, seguridad de la información y de seguridad informática han sido empleados de diversas maneras y se les han dado diversos significados de acuerdo al contexto. Los siguientes párrafos son definiciones que tratan de ilustrar uno de los significados más comunes a cada término”.

a) Seguridad: De acuerdo con el diccionario de la Real Academia Española, seguridad es:
    - Cualidad de seguro.

- Dicho de un mecanismo: Que asegura algún buen funcionamiento, precaviendo que este falle, se frustre o se violente.

b) Seguridad de la Información: Se puede hablar de la seguridad de la información como el conjunto de reglas, planes y acciones que permiten asegurar la información manteniendo las propiedades de confidencialidad, integridad y disponibilidad de la misma.
   - La confidencialidad es que la información sea accesible sólo para aquéllos que están autorizados.
   - La integridad es que la información sólo puede ser creada y modificada por quien esté autorizado a hacerlo.
   - La disponibilidad es que la información debe ser accesible para su consulta o modificación cuando se requiera.

c) Seguridad Informática: Conjunto de políticas y mecanismos que nos permiten garantizar la confidencialidad, la integridad y la disponibilidad de los recursos de un sistema (Por ejemplo: recursos de un sistema como memoria de procesamiento, espacio de almacenamiento en algún medio físico, tiempo de procesamiento, ancho de banda y por su puesto la información contenida en el sistema).

Una vez mencionadas las definiciones anteriores, para que exista seguridad ya sea de la información o informática hay que garantizar las propiedades de confidencialidad, integridad y disponibilidad. Así, es aquí donde se utiliza a la esteganografía, ya que mediante el uso correcto de sistemas esteganográficos se pretende garantizar las propiedades de confidencialidad e integridad.

En la Ilustración 2 se muestra el flujo que debe llevarse a cabo a través de una comunicación normal. En este caso, no existe ningún problema de seguridad informática. El mensaje que se envía se recibe sin alteración alguna.

Ilustración 2 Comunicación normal

El segundo caso, como se muestra en la Ilustración 3; uno de los problemas más grandes que hay, la interrupción de la transmisión del mensaje, que puede ser ocasionada por fallo del canal o de algún elemento del sistema de comunicación, ya sea de forma natural o intencional. Esto es traducido a un problema de disponibilidad.

Ilustración 3 Comunicación con interrupción

Como podemos ver en la Ilustración 4, la interceptación de los datos por un intruso (un intruso es un ente externo al sistema) es algo muy común dentro de las comunicaciones, ya que muchas de las transmisiones son enviadas mediante protocolos que son conocidos por todos y a los mensajes no se les hace ningún tratamiento especial; en otras palabras, viajan tal cual se generan. Lo único que se hace es escuchar todo lo que pasa por el canal sin alterar nada. Este es un problema de confidencialidad.

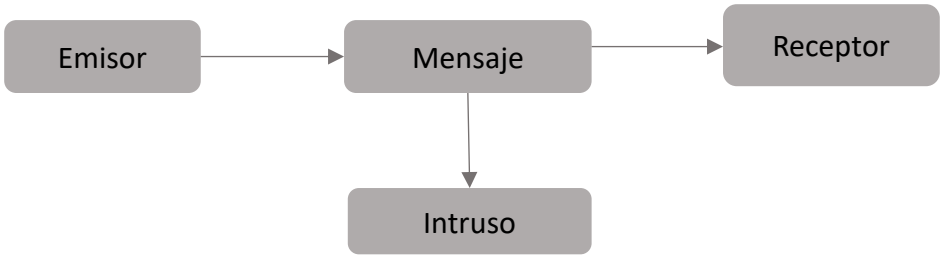


Ilustración 4 Comunicación con intercepción

Otro problema en la comunicación es el problema de la falsificación. Esto se produce cuando el intruso captura un mensaje, se adueña de él y de la identidad del emisor y genera un nuevo mensaje con la identidad del emisor. Este es un problema de integridad y confidencialidad. (Ver Ilustración 5)

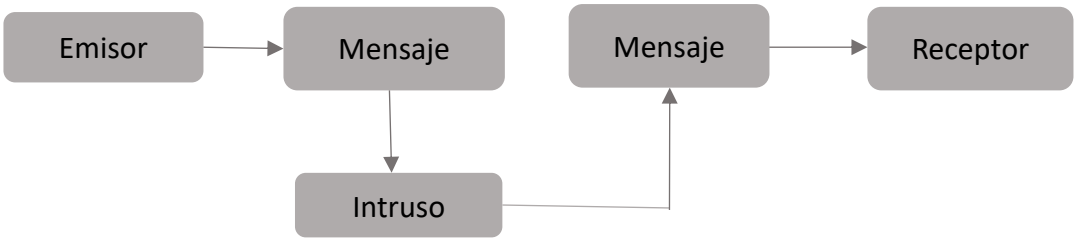


Ilustración 5 Comunicación con falsificación

Finalmente, como se muestra en la ilustración 6 la generación de mensajes se da cuando el intruso genera un mensaje engañando al receptor haciéndolo creer que es un emisor válido. Esto se traduce en un problema de integridad.

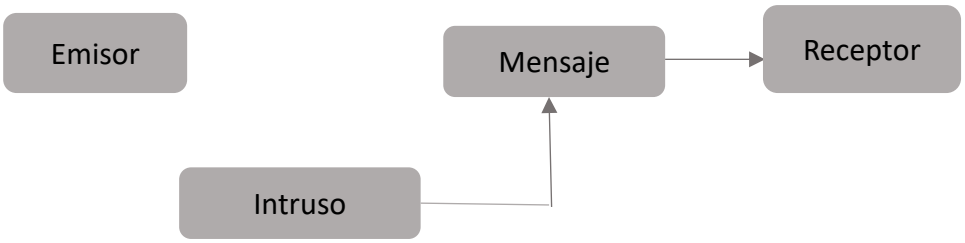


Ilustración 6 Generación de una comunicación apócrifa

Es muy fácil ver como una comunicación y un sistema informático son muy similares, ya que en un sistema informático se procesan, almacenan, envían y reciben datos. Ahora, si pudiéramos de alguna forma evitar los problemas de disponibilidad, integridad y confidencialidad, tendríamos un sistema seguro. Para lograr esto tendríamos que aislar al sistema de los intrusos y hacerlo anti-fallos lo cual es prácticamente imposible. Lo que se hace es crear mecanismos que garanticen en cierta medida las propiedades de disponibilidad, integridad y confidencialidad.

La disponibilidad generalmente se trata de solucionar con sistemas redundantes. La confidencialidad se puede lograr usando un mecanismo que, aunque sea robada la información,

permita que no se pueda acceder a esta o garantice de alguna forma que no se pueda llegar a ella, hasta que pierda su valor.

La integridad es más difícil de lograr y se hace con el uso de varios mecanismos que garantizan la identidad de un ente que está autorizado por el sistema para crear o hacer modificaciones a la información, de tal forma que se puede verificar posteriormente quién creó o modificó la información.

Además, estos mecanismos permiten ver si la información ya creada ha sufrido o no alguna modificación no autorizada. [2]

Los mecanismos para garantizar la integridad y la confidencialidad se pueden implementar con sistemas estenográficos, de ahí la importancia de la esteganografía en la seguridad de la información.

Por lo que surge la necesidad de tener bajo llave sus datos y para esto es necesario conocer de forma detallada el concepto de seguridad de información o seguridad informática partiendo de los 4 niveles básicos de seguridad los cuales se muestran en la Tabla 1.

Nivel	Especificación
Aplicación	Es lo que ve el usuario. Es el nivel más complejo y el menos fiable. La mayor parte de los fraudes informáticos ocurren en este nivel.
Middleware	Implicados en los sistemas de gestión de BD y la manipulación del software.
Sistema operativo	Se trata la gestión de ficheros y las comunicaciones.
Hardware	Es el nivel menos complejo y más fiable. Características de seguridad en el CPU y en el hardware (ejemplo, para evitar desbordamientos de buffer o pila).

Tabla 1 Niveles básicos de seguridad.

2.3 Criptografía

La criptografía es un método cuyo objetivo principal es cifrar y proteger un mensaje o archivo por medio de un algoritmo, usando una o más claves, sin ellas será realmente difícil obtener el archivo original. En nuestros tiempos, la protección de la información cada vez se vuelve una necesidad indispensable. Debido al gran crecimiento y auge de los sistemas informáticos, una gran parte de nuestra vida diaria se rige y ocupa información que se guarda en una computadora. Aún peor, el auge del Internet y de la banda ancha, pone a disposición de una gran cantidad de gente, equipos que contienen información delicada para muchos de nosotros, como direcciones, teléfonos e información financiera entre otras. [3]

En la Internet es relativamente fácil realizar este tipo de engaño, pues uno nunca puede estar seguro de si el correo electrónico realmente proviene del remitente o si nos estamos comunicando realmente con nuestro banco o con otro sitio apócrifo que aparenta ser el banco, por ello es que el uso de algoritmos criptográficos es muy útil. Aun así, la criptografía por sí sola no resuelve todos los problemas, es necesario utilizar toda una infraestructura que le dé fortaleza, para evitar el engaño y asegurar autenticidad e integridad.

2.3.1 Clasificación de la criptografía

La criptografía se puede clasificar históricamente en dos: La criptografía clásica y la criptografía moderna.

La criptografía clásica es aquella que se utilizó desde antes de la época actual hasta la mitad del siglo XX. También puede entenderse como la criptografía no computarizada o mejor dicho no digitalizada. Los métodos utilizados eran variados, algunos muy simples y otros muy complicados. Se puede decir que la criptografía moderna se inició después de tres hechos: el primero fue la publicación de la Teoría de la Información por Shannon; el segundo, la aparición del estándar del sistema de cifrado.

Tanto la criptografía clásica como la moderna se clasifican de acuerdo a las técnicas o métodos que se utilizan para cifrar los mensajes. Esta clasificación la podemos ver en la Ilustración 7.

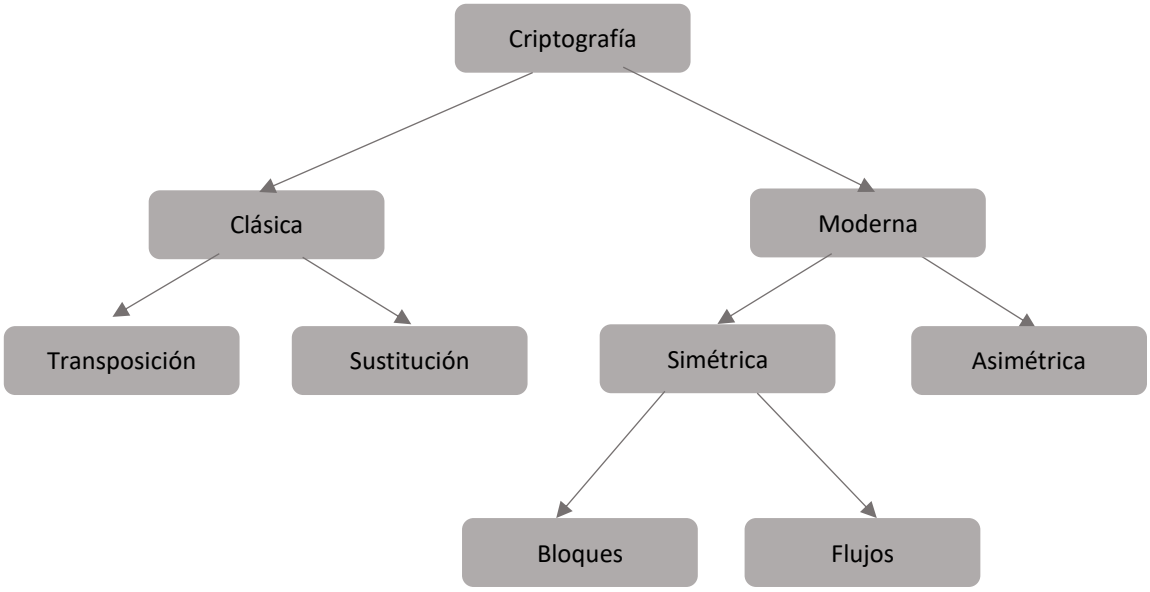


Ilustración 7 Clasificación de la criptografía

2.3.2 Criptografía clásica

Como se mencionó anteriormente, la criptografía clásica es muy antigua. Las técnicas criptográficas eran muy ingeniosas y se usaban para enviar mensajes secretos entre las personas que tenían el poder o en época de guerra para enviar instrucciones. A diferencia de la criptografía moderna, el algoritmo del sistema criptográfico se mantenía en secreto. La criptografía clásica también incluye la construcción de máquinas las cuales, mediante mecanismos, comúnmente engranes o rotores, transformaban un mensaje en claro a un mensaje cifrado como la máquina Enigma usada en la Segunda Guerra Mundial.

Asimismo, se utilizan cifrados por transposición que usan la técnica de permutación de forma que los caracteres del texto se reordenan mediante un algoritmo específico.

Por otra parte, el método basado en sustitución utiliza la técnica de modificación de cada carácter del texto por otro que corresponde al alfabeto de cifrado. Si el alfabeto de cifrado es el mismo que el del mensaje o bien el único, hablamos entonces de cifradores monoalfabéticos; es decir, existe

un único alfabeto en la operación de transformación del mensaje en criptograma. De lo contrario, si en dicha operación intervienen más de un alfabeto, se dice que el cifrador es polialfabético.

Es realmente interesante analizar cada una de las técnicas anteriores, se describirán dos técnicas en este caso: un cifrado de transposición de grupos, la escítala y un ejemplo de sustitución monoalfabética, monográmica con el alfabeto estándar conocido como el cifrado César.

- La escítala

En siglo V a.c. los lacedemonios, un antiguo pueblo griego, usaban el método de la *escítala* para cifrar sus mensajes. El sistema consistía en una cinta que se enrollaba en un bastón sobre el cual se escribía el mensaje en forma longitudinal, como se muestra en la ilustración 8:

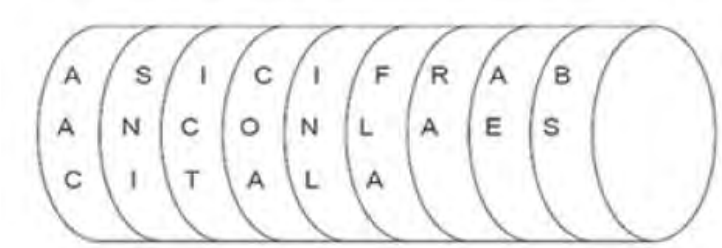


Ilustración 8 Escítala

Habiendo escrito el mensaje, la cinta se desenrollaba y se entregada al mensajero. Para enmascarar completamente la escritura, aunque la cinta debería tener caracteres en todo su contorno. Como es de esperar, la llave del sistema residía precisamente en el diámetro de aquel bastón, de forma que solamente el receptor autorizado tenía una copia exacta del mismo bastón en el que enrollaba el mensaje recibido y, por tanto, podía leer el texto en claro.

- El cifrado César

El cifrado del César aplica un desplazamiento constante de tres caracteres al texto en claro, de forma que el alfabeto de cifrado es el mismo que el alfabeto del texto en claro, pero desplazado 3 espacios hacia la derecha módulo n, con n el número de letras del mismo. A continuación, se muestra el alfabeto y la transformación que realiza este cifrador por sustitución de caracteres para el alfabeto castellano de 27 letras.

Alfabeto	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Alfabeto	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Tabla 2 Alfabeto y transformación del cifrado Cesar

Así con este alfabeto podemos cifrar el siguiente mensaje:

Mensaje original: MENSAJE DE PRUEBA

Mensaje cifrado: OHPVDM GH SUXHED

Al describir el cifrado de César se utilizó un concepto muy usado en las matemáticas y más en criptografía, el módulo.

El módulo es una operación binaria que se realiza en los enteros positivos y se representa de la siguiente forma: $c = a \bmod b$ de tal forma que a, b y c son enteros positivos.
El valor de c al realizar la operación $c = a$ modulo b es igual al residuo de dividir a entre b. Se puede observar claramente que $0 <= c < b$.

Con este antecedente podemos escribir en forma matemática el cifrado de César de la siguiente forma:
Para cifrar

$Ci = (3 + Mi) \bmod 27$

con i= 0,1,…,n ; n = número de letras del mensaje

donde Ci es la letra cifrada y Mi es la letra a cifrar

el alfabeto comienza con A = 0 , B=1, …, Z=26

Para descifrar

$Mi = (Ci - 3) \bmod 27 = (Ci + 24) \bmod 27$

con i= 0,1,…,n ; n = número de letras del mensaje

donde Ci es la letra cifrada y Mi es la letra a cifrar

el alfabeto comienza con A = 0 , B=1, …, Z=26

2.3.3 Criptografía moderna

La criptografía moderna se puede clasificar en dos grandes grupos: la criptografía de llave secreta o simétrica y la criptografía de llave pública o asimétrica.

- Criptografía simétrica:

La criptografía simétrica o de llave secreta es aquella que utiliza algún método matemático llamado sistema de cifrado para cifrar y descifrar un mensaje utilizando únicamente una llave secreta. Se puede observar en la ilustración 9, que la línea punteada es el eje de simetría: lo mismo que hay de un lado existe exactamente igual en el otro, esto ilustra el hecho del porqué se le da el nombre de criptografía simétrica. [1]

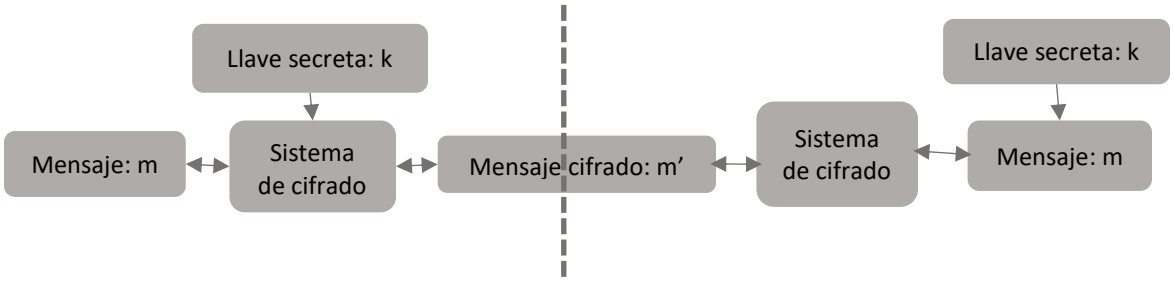


Ilustración 9 Criptografía simétrica

Este tipo de criptografía sólo utiliza una llave para cifrar y descifrar, esto es: si yo cifro un mensaje *m* con una llave secreta *k* entonces el mensaje cifrado resultante *m'* únicamente lo voy a poder descifrar con la misma llave *k*. Este tipo de llave conocida como secreta se debe de compartir entre las personas que se desea que vean los mensajes.

Con este tipo de criptografía podemos garantizar la confidencialidad porque únicamente quien posea la llave secreta será capaz de ver el mensaje.

Criptografía simétrica por bloques:

Este tipo de criptografía está basado en el diseño propuesto por Horst Feistel en los años 70.

- Diseño de Feistel:

Un bloque de tamaño N bits comúnmente N=64 o 128 bits se divide en dos bloques de tamaño N/2, A y B. A partir de aquí comienza el proceso de cifrado y consiste en aplicar una función unidireccional (muy difícil de invertir) a un bloque B y a una subllave k1 generada a partir de la llave secreta. Se mezclan el bloque A con el resultado de la función mediante un XOR. Se permutan los bloques y se repite el proceso n veces. Finalmente se unen los dos bloques en el bloque original.

Algoritmo	Bloque (bits)	Llave (bits)	Vueltas
Lucifer	128	128	16
DES	64	56	16
Loki	64	64	16
CAST	64	64	8
Blowfish	64	Variable	18

Tabla 3 Algoritmos de cifrado de bloque

Criptografía Simétrica de Flujo

Este tipo de criptografía se basa en hacer un cifrado bit a bit, esto se logra usando la operación XOR, representada con Å. Se utiliza un algoritmo determinístico que genera una secuencia pseudoaletoria de bits que junto con los bits del mensaje se van cifrando utilizando a operación XOR.

- Criptografía asimétrica

Si se observa la Ilustración 10, la cual muestra la idea de criptografía de llave pública, se puede ver claramente que no existe simetría en ella, ya que de un lado de la figura se cifra o descifra con una llave pública y en el otro lado con una privada. De este hecho es de donde la criptografía asimétrica debe su nombre. [1]

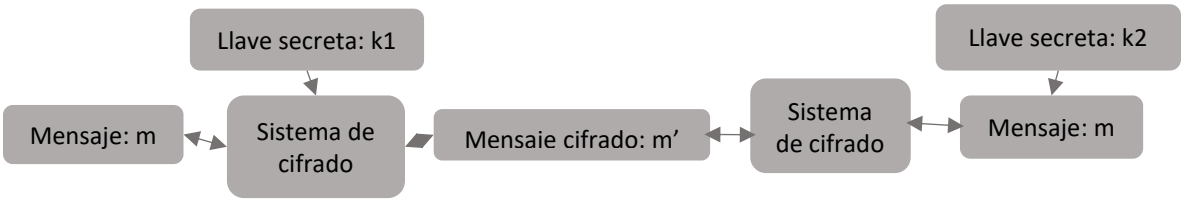


Ilustración 10 Criptografía asimétrica

Es importante destacar que para este tipo de criptografía lo que se cifra con una llave se puede descifrar con la otra llave. Es decir, yo puedo cifrar con la llave pública y descifrar con la privada y

viceversa. Esto es de gran ayuda ya que el número de llaves que debo de poseer se reduce considerablemente.

Proceso criptográfico

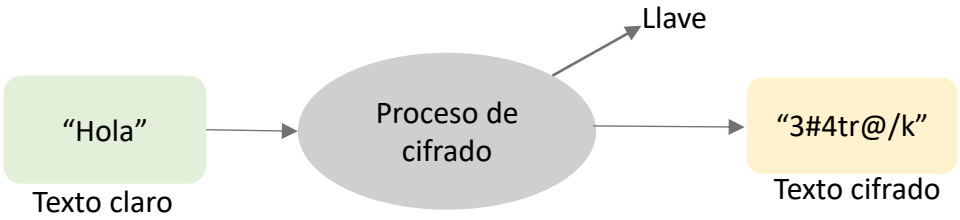


Ilustración 11 Encriptado o cifrado de datos

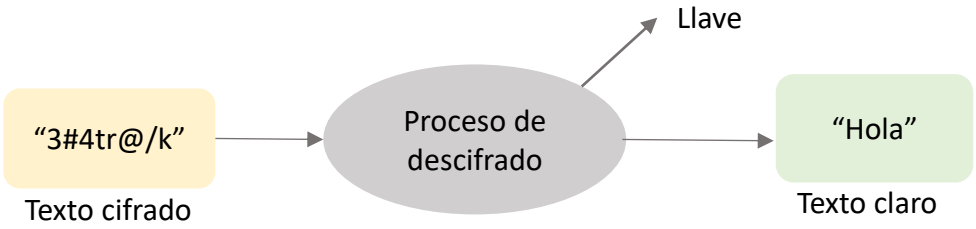


Ilustración 12 Desencriptado o descifrado de datos

2.4 Esteganografía

Del griego steganos (oculto) y graphos (escritura), la esteganografía se puede definir como la ocultación de información en un canal encubierto con el propósito de prevenir la detección de un mensaje oculto.

La esteganografía estudia el conjunto de técnicas cuyo fin es insertar información sensible dentro de otro fichero. A este fichero se le denomina fichero contenedor (gráficos, documentos, programas ejecutables, etc.). De esta manera, se consigue que la información pase inadvertida a terceros, de tal forma que sólo sea recuperada por un usuario legítimo que conozca un determinado algoritmo de extracción de la misma.

Se pueden observar distintos actores implicados en el campo de la esteganografía:

- Objeto contenedor: se trata de la entidad que se emplea para portar el mensaje oculto. Acudiendo al ejemplo de los mensajes sobre el cuero cabelludo, el objeto contenedor es el esclavo en sí.
- Estego-objeto: se trata del objeto contenedor más el mensaje encubierto. Siguiendo con el ejemplo, se trata del esclavo una vez se ha escrito en su cuero cabelludo el mensaje y se le ha dejado crecer el pelo.
- Adversario: son todos aquellos entes a los que se trata de ocultar la información encubierta. El adversario puede ser pasivo o activo. Un adversario pasivo sospecha que se puede estar produciendo una comunicación encubierta y trata de descubrir el algoritmo que se extrae del estego-objeto, pero no trata de modificar dicho objeto. Un adversario activo, además de tratar de hallar el algoritmo de comunicación encubierta, modifica el estego-objeto con el propósito de corromper cualquier intento de mensajería subliminal.
- Estegoanálisis: ciencia que estudia la detección (ataques pasivos) y/o anulación (ataques activos) de información oculta en distintas tapaderas, así como la posibilidad de localizar la información útil dentro de la misma (existencia y tamaño).

Teniendo en cuenta que pueden existir adversarios activos, una buena técnica esteganográfica debe ser robusta ante distorsiones, ya sean accidentales o fruto de la interacción de un adversario activo. [4]

2.4.1 Tipos de esteganografía

Esteganografía pura

Los algoritmos de ocultación y extracción de la información, que sólo el emisor y el receptor del mensaje deberían conocer. Este tipo de esteganografía es seguro siempre y cuando el medio por el cual se transmite el mensaje sea inexperto en las habilidades de esteganografía, pero en medios más avanzados se requiere el uso de este método combinado con criptografía, ya que, si no se combina podría resultar desastroso para el objetivo que se busca.

Un ejemplo claro de esto ocurre en una cárcel, cuando dos prisioneros planean escapar, pero sus notas tienen que pasar necesariamente por el guardia de seguridad quien las revisa antes de entregarlas, su plan lo realizan mediante notas al parecer indefensas, las cuales al ser puestas a contraluz, revelaban el mensaje oculto, este es un claro ejemplo de la aplicación de la

esteganografía pura, cuando el medio que transporta el mensaje, omite los mensajes ocultos que puede contener los ficheros que transporta (ver ilustración 13).

Ilustración 13 Problema de los prisioneros

Esteganografía de clave privada

Tomando en cuenta que un atacante podría conocer los algoritmos de ocultación y extracción de la información. Por lo tanto, el mensaje se cifra utilizando un cifrado simétrico antes de ocultarlo. De esta manera, si el atacante intercepta la transmisión y logra extraer la información aún tendrá que enfrentar el proceso de descifrar el mensaje. Como se mencionó anteriormente para evitar el fracaso a la hora de transmitir mensajes, es necesario agregar otro nivel de seguridad más a el mensaje oculto y es aquí donde la esteganografía gana gran fortaleza frente a otras técnicas de ocultamiento de información, tan sólo se agrega un nuevo parámetro al estego-algoritmo el cual es comúnmente conocido como estego-clave, la estego-clave debe ser socializada entre el emisor y el receptor antes de la comunicación, puede llegar a ser desde que metro de la cinta se debe leer, cada cuántas revoluciones de cassette se debe capturar una letra o incluso que intensidad de luz es la óptima para poder ver el mensaje, de esta manera se tiene infinidad de formas de esconder y es por esto que es casi imposible descifrar el contenido de la comunicación si no se cuenta con la estego-clave.

Esteganografía de clave pública

Por último, este tipo de esteganografía utiliza dos claves y su principal característica es que no requiere un intercambio previo de estego-clave. Requiere de dos claves, una secreta que se utiliza al momento de realizar la inserción del mensaje secreto y una pública, la cual se guarda en las bases de datos públicas. La estego-clave se usa para reconstruir el mensaje.

2.4.2 Propiedades de un sistema de esteganografía robusto

La seguridad de estos sistemas no debe estar basada en la ocultación de los algoritmos utilizados, sino en la fortaleza de los mismos y en la seguridad de la clave.

Entre las propiedades deseables de un sistema de esteganografía se encuentran la robustez, la resistencia a las manipulaciones, imperceptibilidad, el costo computacional y la baja probabilidad de error.

Robustez

Los archivos digitales como imágenes, audio y video, están expuestos a muchos tipos de modificaciones o distorsiones entre las cuales se encuentran las pérdidas por compresión, los cambios producidos por el mejoramiento de imágenes, la amplificación de las señales de audio, etc. Un sistema de esteganografía es considerado robusto si conserva sus características después de esas operaciones en imágenes y video, también se deben conservar estas características después de aplicar alguna técnica o algoritmo esteganográfico. Es decir, que el texto oculto (mensaje) presente en los archivos debe poder ser recuperado después de las distorsiones. Para consolidar su robustez, los sistemas esteganográficos, deben insertar el texto en regiones perceptualmente significativas de los archivos multimedia.

La robustez no debe exigirse incondicionalmente, ya que el sistema de algoritmos puede necesitar ser robusto respecto a determinados procesos y frágil respecto a otros. Si un sistema esteganográfico requiere que ciertas modificaciones de los archivos dañen la información oculta, se le denomina sistema esteganográfico frágil.

Resistencia a manipulaciones

La resistencia a manipulaciones de un sistema esteganográfico es un aspecto que puede relacionarse con la seguridad del mismo; se refiere a su resistencia frente a los ataques hostiles basados en el total conocimiento de los algoritmos de incrustado y detección y de los archivos marcados, excepto de la clave utilizada.

2.4.3 Modelo general de la esteganografía

Para establecer un modelo de comunicación secreta entre dos partes, muchas de las aplicaciones esteganográficas siguen el modelo que se describe en la ilustración 14. En el que el emisor escoge aleatoriamente un objeto que servirá de cubierta (C), el cual puede ser transmitido sin levantar sospecha de ningún atacante.

Posteriormente, un mensaje secreto (M) es oculto dentro del objeto (C) mediante el uso de la estego-clave (K) para dar como resultado final el estego-objeto (S); entonces, el estego-objeto es enviado al receptor haciendo uso de un canal público inseguro.

Luego de que el estego-objeto es obtenido por el receptor, puede recuperar el mensaje oculto, haciendo uso de la estego-clave.

Cabe mencionar que en un sistema como el que se describió, es importante no volver a utilizar el mismo objeto de cubierta para enviar varios mensajes ocultos, evitando así llamar la atención de los atacantes.

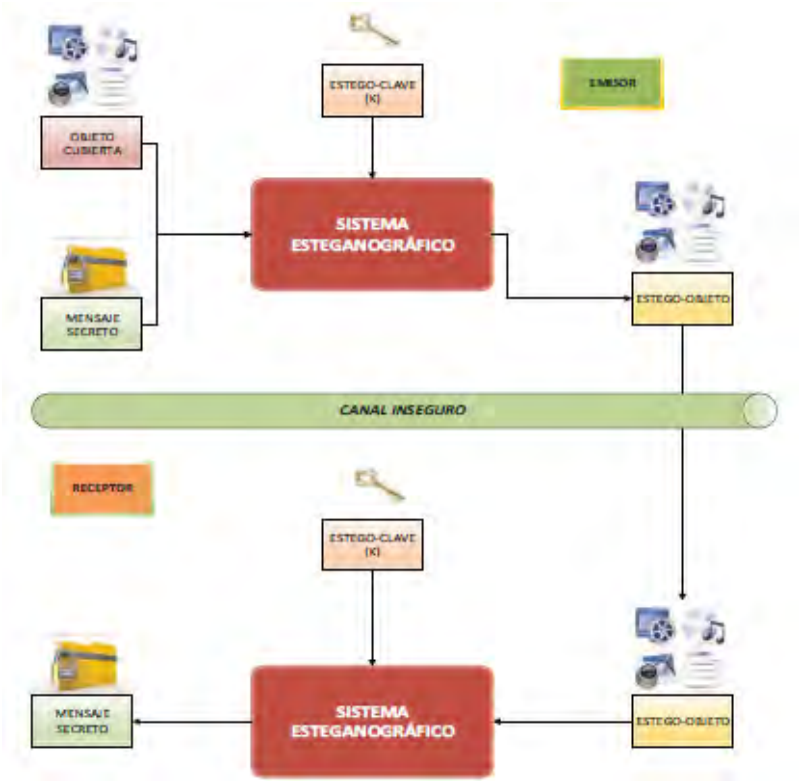


Ilustración 14 Modelo general de esteganografía

2.4.4 Aplicación de la esteganografía

La esteganografía se utiliza con mejores o peores resultados. Veremos algunos de sus usos reales, como pueden ser las marcas de agua, transmisión de mensajes terroristas, detección de copyright y algún que otro uso. Siendo un medio para almacenar información de forma que oculta la existencia de esa información. Tomando en cuenta los métodos de comunicación existentes, la esteganografía puede ser utilizada para realizar intercambios ocultos. Los gobiernos están interesados en dos tipos de comunicaciones: las que apoyan la seguridad nacional y los que no lo hacen. Esteganografía digital proporciona un amplio potencial para ambos tipos. Negocios pueden tener preocupaciones similares con respecto a secretos comerciales o información sobre nuevos productos. Evitar la comunicación en formas bien conocidas reduce en gran medida el riesgo de información que se filtró en tránsito. [5]

La esteganografía de imágenes tiene muchas aplicaciones en distintos ámbitos dentro de esta era digital, por lo que se estable la siguiente clasificación:

- Integridad y autenticación de objetos.
- Protección frente a copias ilícitas.
- Etiquetas, números de serie y huellas digitales
- Diferentes niveles de acceso a datos.

Integridad y autenticación de objetos

Un ejemplo de aplicación de estos procedimientos se encuentra en el campo de la seguridad y vigilancia frente a robos u otros delitos semejantes. Frecuentemente una cámara de video que registra las imágenes del lugar bajo vigilancia, con el propósito de que estas sirvan como prueba en un caso (juicio) pero solo si se demuestra su integridad y autenticidad serán tomadas como válidas.

Ambas propiedades integridad y autenticidad, pueden garantizarse mediante métodos criptográficos, pero también se puede emplear el watermarking o marca de agua digital que es una técnica esteganográfica que consiste en insertar un mensaje (oculto o no) en el interior de un archivo digital, que contiene información sobre el autor o propietario intelectual del objeto digital tratado, la cual tiene como objetivo principal poner el uso ilícito de cierto servicio digital por parte de un usuario no autorizado [6].

- **Marca de agua.**

La idea básica de la marca de agua consiste en insertar información en la imagen mediante la realización de modificaciones sobre la misma, con el objetivo de proporcionar pruebas sobre quién es el propietario de la imagen o a quién ha sido vendida o enviada. La realización de este proceso sobre la imagen deberá ser imperceptible para el ojo humano, sin afectar por tanto a su calidad. El objetivo es facilitar el control que se hace de materiales protegidos por derechos de autor, no sólo garantizar que el acceso a la información es el acordado, sino principalmente que no se va a hacer un uso ilegítimo de la misma; por ejemplo, comerciando con segundas copias. Es en este punto donde supone un importante valor añadido para los autores ya que con los sistemas criptográficos podemos autenticar al comprador y garantizar que el material ha sido entregado a la persona

esperada y sin que nadie haya podido realizar copias ilícitas durante la transmisión. Sin embargo, una vez que el material multimedia está en poder del comprador, éste puede usarlo, copiarlo, revenderlo, etc., sin control por parte del autor. Es aquí donde se detecta un grave problema. [7]

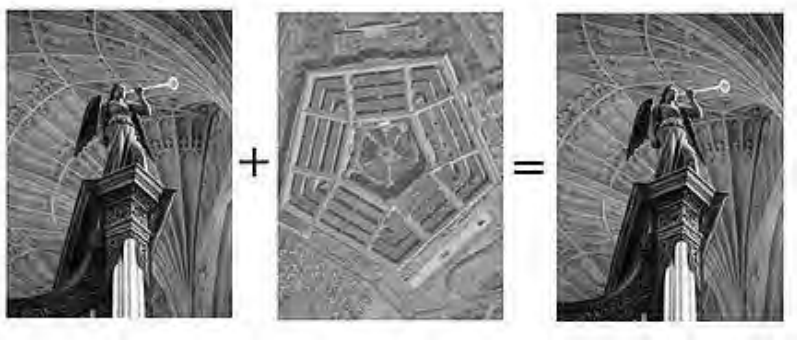


Ilustración 15 Marca de agua

- **Marca de agua sobre papel**

Antes de hablar sobre las marcas de agua digitales y sus aplicaciones en la informática es necesario hablar sobre las marcas de agua sobre papel. Durante el proceso de fabricación del papel y cuando este aún se encuentra húmedo, se emplea una especie de cilindro el cual tiene adherido una rejilla con la señal o la marca que se desea emplear, en muchos casos se utilizan dibujos, escudos o alguna señal que hace distintivo el papel.

En la actualidad esta técnica es usada, por ejemplo, en el papel de los billetes, el objetivo principal de utilizar esta técnica es evitar que los billetes sean falsificados, aunque también se utiliza cuando se desea dar cierta validez a algún libro o estudio impreso en papel además de dotar al libro de información como año en que se elaboró o el lugar de procedencia del mismo.

Ilustración 16 Marca de agua sobre billete de 50 euros

Una de las principales aplicaciones de las marcas de agua que también se conocen como filigranas es en los billetes para evitar que estos sean falsificados, este método consiste en ocultar información ya que estas marcas nos son muy visibles. La marca aparece solo si se observa el billete a contraluz. Se considera es una técnica esteganográfica ya que utiliza un objeto para ocultar información.

- **Marca de agua digital**

La era digital ha llegado y la necesidad de manejar información por la red es importante para cualquier persona o compañía, toda la información almacenada en libros y en documentos importantes tienden a convertirse en archivos digitales. La necesidad de las personas de ser reconocidos como los autores originales de todos estos trabajos es un reto para el cual se proponen las marcas de agua digitales como solución.

Ilustración 17 Copyright: Derecho de copia

Esta técnica se creó como solución a los derechos de copia o copyright de archivos como documentos, imágenes, audio, video. La idea principal de este método es crear una marca que sea inseparable de todos estos archivos donde se pueda conservar información como autor, propietario, distribuidor y evitar el plagio de esta información. Las principales características de esa técnica es que debe ser invisible a la persona que quiera observar su contenido, al momento de implementar esta técnica se debe proteger el contenido que se quiere proteger, es decir, no se debe degradar la información.

Esta técnica consiste en insertar un código directamente al archivo ya sea una imagen, audio o video como lo indica. Este código funciona como un identificador que está asociado al autor, en ocasiones se utilizan otros medios como huellas digitales para reforzar la seguridad del sistema. Este modelo debe ser invisible a terceros que quieran hacer plagio del contenido, pero deber ser accesible mediante generalmente el uso de un algoritmo y una contraseña. [7]

Protección referente a copias ilícitas

La protección de los derechos de propiedad intelectual es probablemente la aplicación más habitual de la esteganografía. Por ejemplo, la disponibilidad de equipos que permiten la realización de múltiples copias de discos compactos ha ocasionado un gran incremento en el mercado de copias ilícitas. Con la finalidad de disminuir este incremento se han diseñado muchos sistemas de protección basados en algoritmos esteganográficos.

Etiquetas números de serie y huellas digitales

En esta área la esteganografía se aplica para conocer las identificaciones tanto del transmisor como el receptor, las cuales serán insertadas dentro de un objeto en este caso una imagen digital. Consiste en números o datos de identificación únicos ocultos en el objeto que se va a proteger, que le brindan al propietario de los derechos de autor, conocer que usuario ha corrompido la licencia de uso proporcionándolo a terceras personas. Estos números se insertan de tal forma que es imposible realizar una copia sin incluir en esta los números de serie, de esta forma se determina la presencia de una copia ilícita y quien la realizo.

Diferentes niveles de acceso a datos

Una última e interesante aplicaciones de la esteganografía es el brindar a los usuarios niveles de acceso múltiples a la información. Estas técnicas permiten la creación de canales ocultos de información accesible solo a determinados usuarios. Por ejemplo, una película difundida en un canal de televisión digital podría incorporar diferentes bandas sonoras en múltiples idiomas.

Programas de ocultación.

Hay que dejar clara la diferencia entre la criptografía y la esteganografía: Cuando solo se utiliza la criptografía, el dato puede ser ilegible, pero es obvio que allí existen datos ocultos. La forma en que actúan juntos criptografía y esteganografía es que la criptografía hace el dato ilegible a quien no conozca la clave y la esteganografía, oculta además la existencia de esos datos. Así los archivos siendo ocultos, hacen que no sea ni leído ni detectado fácilmente. Actualmente, ambas técnicas son perfectamente combinables, complementándose la una a la otra y obteniendo una seguridad aún mayor. Cuando se oculta información en archivos de imágenes se aprovechan los bits menos significativos de los colores para introducir en ellos la información (en la cual se hace una reducción de colores respecto a la imagen original, si es necesario). Si la relación entre la información a ocultar, el tamaño de la imagen y el número de colores es buena, suele ser prácticamente imposible diferenciar la imagen original de la imagen con información oculta. Cuando se usan archivos de sonido, la información oculta aparece como ruido de fondo, pudiendo confundirse fácilmente con una simple grabación con algo de ruido. [7]

Otros métodos de esteganografía más comunes, accesibles y a los que todos hemos tenido acceso sin tener ni idea de que era eso de la esteganografía, son la escritura con tintas invisibles (Por ejemplo: leche, jugo de frutas, vinagre), las cuales al ser expuestas al calor se oscurecen, dejando entrever el mensaje oculto. Igualmente, sencillo e inocente resulta enviar un mensaje escribiendo un texto, y luego, tomando la primera letra de cada palabra, se podía leer el mensaje que en realidad se quería transmitir.

En resumen:

La esteganografía es el arte de transmitir información de modo que la presencia de la misma pase inadvertida, típicamente escondida dentro una imagen, archivo de audio o en su caso video digital. [7]

Esteganografía clásica consiste en métodos completamente obscuros es una protección basada en descomponer el canal encubierto especifico que se está usando.

Esteganografía moderna consiste en el uso de canales digitales como por ejemplo archivos de texto, audio digital, imágenes, video, protocoles de comunicaciones (TCP/IP).

2.4.5 Esteganografía en imágenes, audio y video digital

Las imágenes digitales, el audio y el video digital, se han postulado como los estegomedios más propicios para ocultar información. En los últimos años se ha publicado múltiples técnicas esteganográficas centradas en este tipo de contenidos [7], así como nomenclatura en función de cómo se trabaje con la información de estos formatos. Los diferentes procedimientos esteganográficos se clasifican en función de las transformadas o criterios matemáticos que se utilizan para la ocultación de información. Por ejemplo, son famosas las técnicas que trabajan en el dominio de la señal, dominios transformados como la transformada discreta de Fourier, la transformada discreta del coseno (por ejemplo, en ficheros JPEG), transformada Mellin-Fourier, transformada discreta Wavelet y sus coeficientes etc. Independientemente de todos estos conceptos, lo que debe tenerse en cuenta para avanzar en procedimientos esteganográficos en contenidos multimedia es conocer el formato del fichero sobre el que se desea trabajar y acceder a las posiciones donde se almacena la información útil, aquella información que puede ser utilizada con fines esteganográficos.

En unos casos, serán pixeles, en otro caso coeficientes de transformadas matemáticas (por ejemplo, ficheros JPEG), etc., Que portador y qué transformada utilizar dependerá de la aplicación que se desee, la seguridad esperada y la invisibilidad requerida.

Es común que en algunas técnicas que trabajan en el dominio espacial proporcionen mayor capacidad de ocultación, mientras que los del dominio transformado (por ejemplo, dominio de la frecuencia) sean más robustos contra ataques, tales como compresión, o procesamiento, en general, de la señal.

Aunque podrían desarrollarse procedimientos de ocultación basados en cualquiera de las técnicas generales de ocultación mencionadas anteriormente, en la práctica las técnicas de ocultación más utilizadas en formatos multimedia son técnicas de inserción que modifican el fichero fuente para ocultar unos datos. En general, la información añadida actuará a modo de ruido adicional que puede provocar perturbaciones en el archivo digital generado. Por este motivo, es común que las técnicas de ocultación se aprovechen de las limitaciones del sistema de percepción humano (vista y oído) para minimizar que un usuario pudiera darse cuenta a simple vista de la presencia de información oculta.

Por ejemplo, la modificación de los colores de una imagen en una región donde hay presente una variedad de colores es menos apreciable que si se modifica el color de ciertos píxeles en regiones uniformes con algunos colores. Del mismo modo, el ojo humano es muy sensitivo a modificaciones en los bordes de una imagen (en el contorno).

A continuación, se van a analizar de forma resumida algunos aspectos interesantes de la ocultación de información de formatos digitales de video, audio e imágenes.

- Las imágenes digitales

Los principios en los que se fundamentan las técnicas de ocultación sobre este tipo de archivo digital son dos: que la modificación de la imagen no introduzca un ruido visual que levante sospechas a una persona que vea la imagen y que las modificaciones introducidas no proporcionen pistas adicionales a un intruso. Existe una gran variedad de técnicas para diferentes formatos gráficos de fichero (bmp, gif, jpeg, png, etc.)

Las técnicas y variantes más documentadas de estos procedimientos consisten en la modificación de los LSB (Least Significant Bit) de los píxeles de una imagen, de los índices que enlazan a la paleta de colores de un formato GIF (otros procedimientos como el reordenamiento de los colores de la paleta es posible) o de los coeficientes resultantes de aplicar alguna transformación matemática a una imagen, por ejemplo, los coeficientes DCT (transformada discreta del coseno) del formato gráfico JPEG o los coeficientes Wavelet del formato gráfico JPEG2000.

Posteriormente se va analizar resumidamente algunas de las técnicas más famosas para los formatos de ficheros BMP, GIF y JPEG. En general, cualquier fichero con formato gráfico podrá ser utilizado con fines esteganográficos.

- Esteganografía en video digital

Una opción interesante para ocultar información son los formatos digitales de vídeo debido a que presentan un volumen considerable de datos que se pueden modificar para ocultar información. En 1998, *A.* Westfeld y G. Wolf [7] demostraron cómo un sistema real, en concreto de videoconferencia, se podía utilizar para establecer un canal oculto de información sin que la señal sufriera una degradación grande. En general, un video puede definirse como un conjunto de marcos (frames) individuales formado por imágenes y audio (y en ocasiones también texto). Los procedimientos de ocultación de información en un vídeo se pueden aprovechar de las distintas técnicas esteganográficas sobre cada uno de esos elementos individuales (estegomedios) que configuran cada marco, algunos de los procedimientos específicos documentados en videos han sido: codificación de información a partir del cálculo de los vectores de movimiento entre una colección de marco, técnicas basadas en corrección de errores, etc.

- Esteganografía en audio digital

En la actualidad el avance de la tecnología, ha permitido la creación de dispositivos cada vez más pequeños y autónomos, que facilitan la creación, reproducción, replicación y distribución de audio. Dispositivos cotidianos, portátiles o no, que permiten estas tareas son, por ejemplo, los ordenadores personales, PDAs, móviles, reproductores de diferentes soportes y formatos de audio (lectores de CD/DVD, MP3/MP4, iPod, audiostreaming, radios-online personales, etc.), dispositivos sintonizadores de radio (FM/AM), grabadoras de voz, software informático de procesamiento de

audio, etc. Por no hablar de los clásicos instrumentos musicales y los soportes para compartir dicha información, alejadas de un mundo claramente digital, como son las clásicas partitura musical.

El estudio de las limitaciones del sistema de audición humano es el punto de partida para el diseño correcto de un algoritmo esteganográfico que oculte información en señales de audio [8]. El sistema de audición humano es bastante más difícil de engañar que otros sentidos, por ejemplo, que el sistema de visión.

El oído presenta una sensibilidad alta a la presencia de un ruido blanco gaussiano añadido a una señal de audio. Esta detección puede ser incluso de unos 70dB por debajo del nivel de ruido ambiente. Sin embargo, existen diversas situaciones en las que el sistema puede ser engañado, es decir, es impreciso en la detección. Una de estas situaciones consiste en que ante la presencia de sonidos fuertes y sonidos más débiles los primeros tienden a enmascarar a los segundos. Por otra parte, el sistema de audición humano es poco sensible a ciertos cambios de fase en una señal de audio, o a la supresión de ciertas frecuencias en la señal de audio, modificaciones en las que se basa el estándar MPEG-1 audio Layer III (mp3).

Existen diferentes procedimientos esteganográficos y estegoanalíticos en señales de audio: técnicas basadas en LSB (Least Significant Bit) en muestras de audio, técnicas de ocultación en la fase de una señal (modulación de la fase de una señal y codificación en la fase), técnicas de ocultación en el eco de una señal, ocultación aprovechando las características estadísticas de las señales de audio (por ejemplo, segmentación de la señal de audio de forma adaptativa), ocultación basada en algoritmos de compresión (MP3, WMA, OGG), etc.

Sin embargo, al igual que sucede con otros portadores, que la introducción de modificaciones puede crear patrones no comunes que pueden ser detectados, por ejemplo, mediante estudios estadísticos de las propiedades de la señal empleada, en muchos casos varios ataques serían válidos para distintos formatos.

2.4.6 Imágenes digitales

El componente más pequeño de una imagen digital se llama pixel, cada pixel se codifica mediante un conjunto de bits de longitud determinada. Las longitudes más empleadas son: 8 bits, usado por

las imágenes en escala de grises, con lo que un pixel puede tomar un valor entre 0 y 255, 24 bits usado para las imágenes a color o con 48 bits usado por imágenes de alta resolución [6].

Una imagen digital tiene diferentes parámetros que la definen, entre los principales se encuentran los siguientes:

- Resolución
- Dimensiones de la imagen
- Profundidad de bits
- Formato de imagen

Resolución de una imagen digital

La resolución de una imagen es el indicador de la cantidad de información que hay en pixeles o puntos de impresión, en un área determinada ya sea en centímetros o en pulgadas. Las unidades más utilizadas son puntos por pulgada, cuando se refiere a una imagen impresa, o pixeles por pulgada, cuando se refiere a una imagen que será mostrada en una pantalla digital. Las medidas estándar más comunes usadas para resolución de imágenes digitales son 72 ppi (pixels per inch) cuando se refiere a visualización de imágenes en pantalla.

En resumen: la resolución de una foto es expresada en pixeles por pulgada (ppi) y en impresión es expresada en puntos por pulgada (dpi).

La Ilustración 18 muestra una comparación en la cual se puede apreciar la calidad de la imagen en distintas resoluciones.

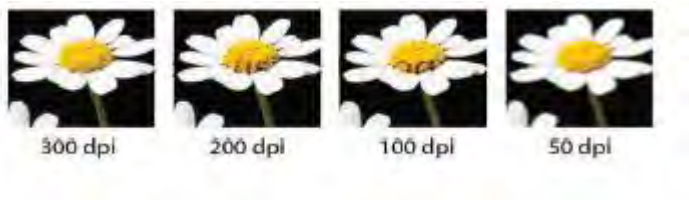


Ilustración 18 Imagen a diferentes resoluciones

Dimensiones de una imagen digital

Se refiere a las medidas horizontales y verticales de la imagen expresada en pixeles. Se pueden determinar realizando la multiplicación del ancho y de la altura de la imagen, medida en pulgadas, por la resolución puntos por pulgada.

Profundidad de bits

La profundidad de bits de una imagen digital se determina por la cantidad de bits que se utilizan para representar a un pixel y se mide en bits/pixel (ver ilustración 19).

A medida que aumenta la profundidad de bits, se puede disponer de una mayor gama de colores. Las imágenes digitales se pueden representar a blanco y negro, escala de grises o a color.

Una imagen en blanco y negro cuenta con un bit para representar a cada pixel en donde el 1 representa el color blanco y el 0 el color negro.

Una imagen en escala de grises puede representar a un pixel con 2 hasta 8 bits, generalmente el valor de cada pixel equivale a una tonalidad diferente de color gris, empezando desde el negro más profundo hasta alcanzar el color blanco.

En las imágenes a color se puede representar un pixel con una profundidad entre 8 y 24 bits o más, dependiendo de los detalles de color que se deseen visualizar en la imagen. Por ejemplo, en una imagen que utilice 24 bits de profundidad y el modelo de color RGB (Red-Green-Blue), usará 8 bits para el color rojo, 8 bits para el color verde y 8 bits para el color azul, los demás colores se obtendrán al realizar combinaciones de estos bits.

Ilustración 19 Profundidad de bits de izquierda a derecha: imagen binaria de 1 bit, a escala de grises de 8 bits y a color de 24 bits.

Compresión

Se utiliza para reducir el tamaño del archivo de imagen y de esta manera, facilitar su procesamiento, almacenamiento y transmisión.

Básicamente es reducir o eliminar las componentes de color redundante, existente en una determinada área de la imagen, para esto generalmente se utiliza la codificación Huffman.

Los sistemas de compresión se caracterizan en compresión sin pérdidas y compresión con pérdidas.

La compresión sin pérdidas, representa cierta cantidad de información en un espacio menor, siendo posible la recuperación exacta de la información

En la compresión con pérdidas, se representa cierta cantidad de información, usando una menor cantidad de la misma, siendo imposible realizar una reconstrucción exacta de la información original.

Formatos de imagen digital

La mayoría de formatos de imágenes digitales se componen de una cabecera que contiene atributos, por ejemplo, las dimensiones de la imagen, el tipo de codificación, etc.

A continuación, se detallan los formatos de imágenes digitales más comúnmente utilizados:

- Windows BitMaP (BMP): este formato de imagen consiste en una cabecera, seguida de los valores de cada pixel de la imagen (cada pixel puede tomar valores de 4, 8, 16, 24 y 32 dependiendo de la calidad de la imagen). Es el formato de imagen más simple y aunque soporta compresión de imágenes no es aplicada. Dado que no se aplica ningún tipo de compresión, las imágenes almacenadas con este formato ocupan un tamaño considerable y no son recomendadas para su transmisión a través de una red de comunicaciones.
- Graphics Image Format (GIF): es un formato de compresión excelente para imágenes con grandes áreas homogéneas de color, y es muy sencillo para realizar animaciones vectoriales. Para la compresión utiliza el algoritmo LZW (Lempel-Ziv-Welch). La principal desventaja que posee es que utiliza 8 bits para la representación de cada pixel, con lo que limita su paleta de colores a tan solo 256 posibilidades y lo hace un formato muy malo para imágenes que requieren alta definición en sus detalles.
- Joint Photographic Experts Group (JEPG): es el algoritmo de compresión más popular debido a que se aprovecha de una limitación del ojo humano, la cual impide la completa visualización de la paleta de colores de 24 bits, razón por la que elimina información que el opjo humano no es capaz de procesar. Esto produce una compresión considerable de la imagen digital, pero produce perdida de información.
- Portable Network Graphics (PNG): este formato de compresión resulta debido a la necesidad de sustituir a GIF, ya que se presentaron problemas con la parte del algoritmo (LZW). PNG cubre en su totalidad todos los aspectos de GIF, utilizando un algoritmo de compresión mejor y con una paleta de colores de 2^{16} posibilidades, superior a la usada por GIF. [6]

2.4.7 Clasificación de las técnicas utilizadas para la esteganografía

Se han desarrollado algunas técnicas para ocultar información, algunas con cierto grado más de dificultad que otras, pero de manera general se pueden clasificar de acuerdo al tipo de cubierta que se usa para la comunicación secreta o de acuerdo a las modificaciones que se hacen a la cubierta del proceso de inserción.

Técnica de sustitución LSB (Least Significant Bit)

La técnica LSB es el mecanismo de ocultación más utilizado en esteganografía. Su utilización es muy común, debido a su facilidad de aplicación, sobre todo a imágenes y audio, permitiendo ocultar grandes cantidades de información. Para comprender su uso vamos a analizarla en primera instancia en uno de los formatos gráficos más sencillos.

Una imagen digital se puede estudiar como un conjunto de unidades de tamaño mínimo, denominados píxeles, que tienen unos determinados valores que reflejan los diferentes colores visibles para una imagen concreta. Dependiendo de la resolución de la imagen y de la codificación empleada para un formato gráfico concreto, los píxeles se representan con 1 o más octetos en dicho formato. Es habitual encontrarse ficheros gráficos con diferentes codificaciones, por ejemplo, ficheros gráficos BMP (BitMaP) con resoluciones 8, 16, 24 o 32 bits. Para una imagen con una resolución de 24 bits cada píxel se representa con 3 octetos (8 bit x 3 = 24 bit). Si estos octetos representan un modelo de color RGB (Red, Green, Blue), cada octeto, respectivamente, almacenará información, sobre el nivel de rojo, verde y azul, que está presente en cada píxel. Esta información permite obtener el color real de un píxel.

El valor concreto que puede tomar cada octeto representa la intensidad de dicho color. Este valor oscila desde el valor 0 (el nivel más oscuro) a 255 (el nivel más luminoso). Por ejemplo, un píxel representado con los siguientes 3 octetos: 11111111 (rojo) 00000000 (verde) 00000000 (azul) da como resultado un píxel de color rojo. Siendo estrictos, la técnica LSB aplicada a imágenes es un procedimiento de sustitución que permite modificar el bit menos significativo de la codificación de cada píxel de una imagen por el bit del mensaje a ocultar.

Técnicas esteganográficas basadas en paleta de colores

Los cambios sobre el tratamiento y almacenamiento de imágenes, dio lugar a la necesidad de representar las imágenes en formatos de ficheros gráficos, de tal forma que su tamaño final, en octetos, fuera no excesivamente grande. Lo cual facilitaría no solo su almacenamiento sino también su intercambio por las redes de telecomunicaciones. Para lo cual se propusieron varias opciones, entre ellas, destacó la idea de crear imágenes con un número pequeño de colores. Si el

número de colores en una imagen es pequeño se pueden codificar estos valores en una especie de tabla, de tal forma que cada píxel en lugar de almacenar el color que le corresponda apunte a su color presente en esa tabla. El tamaño con el cual se codifica estos índices que apuntan a la tabla es menor que la codificación con la que se almacena el color para cada píxel, con lo que se consigue, finalmente, ficheros gráficos de tamaño más pequeño. Esta ordenación especial, de los octetos de la imagen, permite reducir bastante su tamaño. Esta información adicional, a modo de tabla, se conoce como paleta de colores.

Un ejemplo de uso de paleta de colores puede verse en el formato gráfico GIF aunque también es posible encontrarla en otros formatos gráficos, por ejemplo, en los ficheros BMP. La paleta de colores de los ficheros GIF permite representar hasta 256 colores diferentes, codificando el valor de cada uno con 3 octetos, que indican su valor RGB. Los valores RGB de los píxeles de la imagen no se almacenan directamente en el fichero gráfico, sino que se almacenan índices que enlazan a la paleta de colores. De esta forma, en un fichero GIF, cada píxel se puede representar con 8 bits (28=256 colores posibles de la paleta) pudiendo servir de índice o entrada para indicar cuál es el color real del píxel que está en la paleta.

Una comparación rápida entre este formato y otros formatos gráficos más clásicos se puede ver con imágenes en formato BMP con una resolución de 24 bits. Para esta resolución, cada píxel necesitaría 3 octetos para codificar el valor del color que representa (su valor RGB). Independientemente de la mejor conveniencia de utilizar uno u otro formato gráfico que contenga una imagen, es cierto, que estos formatos que se apoyan en la utilización de una paleta de colores facilitan la creación de diferentes técnicas de ocultación de información.

Ilustración 20 Ejemplo de paleta de colores con 3 colores e imagen de 6 píxeles.

Esta técnica es fácil de comprender en el caso del formato GIF (como se muestra en la Ilustración). En este formato los datos de la imagen son octetos individuales. En lugar de que el octeto contenga información de los colores del pixel indica la posición en la paleta de colores donde se encontrará el valor real del pixel. Así, por ejemplo, un octeto con valor 0 apunta a la entrada 0 de la paleta de colores, un octeto con valor 1 apunta a la entrada 1, y así sucesivamente para los 256 valores posibles que puede tomar cada octeto. En el ejemplo de la Ilustración 11 el valor "real de los píxeles de la imagen" sería: color verde (3 octetos) apuntado por el índice (octeto) con valor 1, color rojo (apuntado por el índice con valor 0), color azul (apuntado por el índice con valor 2), etc. Es habitual que se modifique el bit menos significativo de la codificación de cada octeto de los datos de la imagen. Si modificamos algún bit a un índice su valor cambiará, por tanto, ese nuevo índice apuntará a otro color diferente presente en la paleta. Aunque esta técnica es fácil de utilizar debe tenerse en cuenta una serie de consideraciones. Al modificar el bit menos significativo de un índice (ponerlo a 0 o 1) el color actual que corresponde al píxel en cuestión cambia, ya que el nuevo índice apuntará al color adyacente del actual, presente en la paleta de colores. Este hecho hace que si los colores adyacentes en la paleta no son parecidos el "ruido" debido a la manipulación de los LSB será obvio, y producirá una imagen la cual tendrá un resultado visual que hará sospechar una posible ocultación de información.

Para intentar reducir este problema es común que antes de insertar la información a ocultar se realice algún procesado u ordenación previa de los colores presentes en la paleta de colores. Sin embargo, este hecho debe realizarse con precaución ya que puede crear patrones inusuales que no se suelen encontrar en los ficheros gráficos. Por ejemplo, es habitual que los valores de la paleta de colores estén ordenados del más usado al menos usado, para reducir tiempos de decodificación de la imagen, o que los colores de esta no presenten cambios graduales (por ejemplo, de un bit), salvo ciertas excepciones, como son las imágenes de grises.

Técnicas esteganográficas basadas en coeficientes. JPEG

En la actualidad el procesamiento digital y almacenamiento de imágenes, ha hecho que se utilicen un sinfín de algoritmos y transformaciones matemáticas con diferentes propósitos, entre ellos, su aplicación a la esteganografía, especialmente los procedimientos de ocultación mediante transformaciones en el dominio de la frecuencia. En concreto, la ocultación de información

utilizando los coeficientes cuantificados presentes en un fichero gráfico con formato JPEG, obtenidos al aplicar a una imagen original una transformada discreta del coseno (DCT). Esta técnica es la más difundida para ocultar información en imágenes en formato JPEG.

Las investigaciones se centran fundamentalmente en la utilización de los coeficientes cuantificados DCTs con fines esteganográficos. El formato JPEG es fácil comprender cuál es el significado real de estos coeficientes cuantificados y por qué pueden utilizarse en esteganografía. Los coeficientes cuantificados, son números que almacenan la información de la imagen a visualizar.

En este caso se aplica una técnica LSB de forma secuencial e individualizada a los coeficientes cuantificados DCTs de la imagen, de modo que para ocultar un bit de información se modifica el bit menos significativo del valor de un coeficiente cuantificado. Esta herramienta tiene la ventaja de proporcionar una alta capacidad de almacenamiento de mensajes ocultos, así como resistencia frente a ataques publicados, como los ataques visuales. Sin embargo, la modificación de los bits LSB de los coeficientes cuantificados introduce anomalías que son detectables por estudios estadísticos del histograma.

Sistema de dominio de la transformada.

Estos sistemas tratan de ocultar información secreta en un área significativa de la imagen que sirve como cubierta, con lo cual son más robustos los ataques como compresión o adición de ruido. Entre los métodos más comunes está el usar la transformada (DTC) o también está el uso de la transformada de wavelet.

2.5 Relación entre esteganografía y criptografía

Para un atacante que desee emitir mensajes a nombre de otro, en el caso criptográfico, la seguridad radica en la fortaleza de la clave privada del emisor atacado, mientras que en el caso esteganográfico, la seguridad radica en el secreto de la aplicación de esteganografía para transmitir el generador del mensaje. Romper el esquema de autenticación implica vulnerar cada uno de estos aspectos. [9]

La criptografía intenta proteger el mensaje de un enemigo, pero sin esconder que se realiza una comunicación y una protección a la misma. Mientras que con esteganografía, se intenta ocultar que se protege la comunicación. Por lo tanto, se pueden enumerar las consecuencias de esta diferencia como las siguientes:

- La existencia de un mensaje cifrado, con aplicación de criptografía, levanta sospecha al enemigo que lo observa: el cual podría amenazar o torturar a los participantes de la comunicación legítima para que revelen el contenido del mensaje. Con la aplicación de esteganografía no se levanta sospecha de una protección a la comunicación.
- Encriptar mensajes alienta desarrollar mecanismos de extorsión para conseguir el mensaje oculto.
- El hecho de encontrar un mensaje encriptado, suele alentar el desafío de saber cuál es el mensaje y esto le motiva a encontrar mecanismos para romper la seguridad y obtener el mensaje original.
- La detección de un mensaje encriptado puede ser suficiente para un enemigo: en ocasiones no es necesario conocer el contenido del mensaje. El sólo hecho de conocer que se efectúa una comunicación puede ser suficiente información.
- El procesamiento de las computadoras aumenta y atenta permanentemente contra las características de los algoritmos criptográficos: un algoritmo seguro hoy en día puede no serlo en un futuro. Si la seguridad se basa en la clave criptográfica, un ataque por fuerza bruta será más rápido de desarrollar con equipos computacionales con mayor capacidad de procesamiento.
- Suele ser difícil demostrar la confiabilidad de algoritmos criptográficos y, en ocasiones, la vulnerabilidad de la seguridad criptográfica puede radicar en el algoritmo utilizado.
- Por último, puede mencionarse una popularidad ampliamente mayor de la Criptografía respecto de la Esteganografía. Esto puede deberse a que la aplicación de técnicas criptográficas es en la práctica más fácil que las técnicas esteganográficas. En particular las técnicas esteganográficas requieren una alta sincronización entre el emisor y el receptor para mantener la comunicación.

2.6 Programación en Matlab

Matlab es una herramienta informática que surgió para realizar cálculos matemáticos, especialmente operaciones con matrices. Además de realizar cálculos, esta herramienta permite crear gráficos de muchos tipos y presenta grandes ventajas a la hora de trabajar con números complejos, con matrices, con polinomios, con funciones trigonométricas, logaritmos, etc.

La programación se lleva a cabo mediante un lenguaje que es muy parecido a lenguajes de alto nivel como Basic o C. esto nos permite que el usuario pueda agrupar sentencias que son utilizadas de manera frecuente en un programa las cuales pueden ser invocadas posteriormente.

2.6.1 Entorno

El programa Matlab se maneja escribiendo sentencias dentro de una ventana llamada de comandos. Las órdenes se escriben una a una pulsando la tecla del retorno al final. Por ejemplo, si se escribe sqrt(16) el programa realiza la operación indicada y responde en la pantalla con el resultado. Lo que se muestra en la pantalla de ordenas será algo como:

```
>> sqrt(16)
ans =
    4
>>
```

Las operaciones se indican de forma muy intuitiva, por ejemplo para obtener el resultado de $\frac{13*5}{3+4*11}$ únicamente es necesario escribir 13*5/(3+4*11). De tal manera que en la ventana de órdenes aparecerá lo siguiente:

```
>>13*5/(3+4*11)
ans=
        1.3830
```

Es necesario utilizar los paréntesis para que la operación se lleve a cabo correctamente.

2.6.2 Vectores

Los vectores se introducen en Matlab como una colección de valores. Por ejemplo, un vector fila es:

```
>> v = [-1 2 -3 4 -5]
```

```
V=
        -1   2   -3   4   -5
```

Para acceder a los componentes individuales del vector se usan paréntesis indicando el índice de la componente como se muestra en las dos sentencias del ejemplo siguiente:

```
>>v(3)
ans =
         -3

>> v(1)*8
ans =
        -8
```

Con esto podemos deducir que en Matlab la expresión $v(k)$ equivale a la notación matemática v_k.

También se puede construir vectores columna. Para ello se separan las filas mediante punto y coma como se muestra a continuación.

```
>> u=[0; 1; 0; 1; 0]
u =
  0
  1
  0
  1
  0
```

Los vectores son tratados por Matlab como matrices con la peculiaridad de tener una sola fila o columna.

2.6.3 Matrices

En Matlab una matriz es una variable como cualquier otra y no precisa mecanismos complicados para su creación y uso. Ejemplo: considere que se desean sumar las siguientes dos matrices:

$$A=\begin{pmatrix}0 & 2 & 4\\ 1 & 1 & 1\end{pmatrix} \qquad B=\begin{pmatrix}3 & 3 & 3\\ -1 & -1 & -1\end{pmatrix}$$

Lo primero que debemos hacer es introducir las matrices en variables. Comenzando con la matriz A escribiendo la sentencia de la siguiente manera:

```
A = [0 2 4; 1 1 1]
```

En la pantalla se obtiene lo siguiente:

```
>> A = [0 2 4; 1 1 1]
A =
```

$$\begin{matrix} 0 & 2 & 4 \\ 1 & 1 & 1 \end{matrix}$$

>>

Como podemos ver, los elementos de una misma fila se separan por espacios o comas y una fila se separa de la siguiente forma mediante el punto y coma.

Se procede del mismo modo con la matriz B, obteniendo lo siguiente:

>> B = [3 3 3; -1 -1 -1]

B =

$$\begin{matrix} 3 & 3 & 3 \\ -1 & -1 & -1 \end{matrix}$$

2.6.4 Cadenas de caracteres

Las cadenas de caracteres son un conjunto de símbolos tomados de la tabla ASCII. Tienen gran utilidad para trabajar en problemas donde se precisa manipulación de texto.

Un ejemplo de cadena de caracteres es: “¡clase de 3 a 5 atrasada!” Este conjunto de caracteres incluye letras, números y signos como el espacio o la admiración. La forma en que la cadena se introduce en Matlab es mediante una variable, quedando de la siguiente manera:

>>texto = ‘¡clase de 3 a 5 atrasada!’

texto =

¡clase de 3 a 5 atrasada!

Como podemos notar la cadena debe ir encerrado por apóstrofes. Son las apóstrofes las que hacen notar la diferencia entre una función o variable de una cadena. Por ejemplo, la diferencia entre las sentencias a= barco y a=’barco’. A continuación, el resultado de cada una de estas:

Caso 1:

>> a=barco

Undefined function or variable 'barco'.

Caso 2:

>> a = 'barco'

a =

barco

>>

En el caso 1 Matlab muestra en la pantalla un mensaje de error pues ha interpretado a=barco como una asignación en la cual el término de la derecha (variable barco) no existe. A diferencia en el caso 2 la expresión, 'barco' es un valor definido que asigna a la variable `a` por lo que la expresión es tan correcta como si hubiera ingresado x= 8.
Las cadenas de caracteres no sirven para cálculos matemáticos, sin embargo, fueron introducidas en Matlab para facilitar la programación, depuración y el uso de programas [10].

Manipulación de imágenes como matrices en Matlab:
Matlab es un software matemático que ofrece un entorno de desarrollo integrado con lenguaje de programación propio.

Está orientado para llevar cabo computaciones numéricas casi de todo tipo, pudiendo manipular vectores y matrices tanto reales como complejas con funciones y fórmulas de varias ramas de la matemática. Matlab se compone de un programa básico y un conjunto de cajas de herramientas para labores más especializadas. [14]

La ventaja de operar imágenes con Matlab, es que se cargan al entorno de trabajo en forma de matrices, y dado que este software está orientado a aplicar cálculos para matrices, es que resulta una herramienta factible para el procesamiento matemático; además de que las imágenes se procesan digitalmente a través de la caja de herramientas (Toolbox) de procesamiento de imágenes con el que cuenta (por ejemplo, aplicar filtros, máscaras, etc.).

2.7 Algoritmo LSB

Un algoritmo representa el planteamiento de una posible solución a un problema Cuando un algoritmo es implementado en cualquier lenguaje de programación, reflejando las ideas desarrolladas en la etapa de análisis y diseño del algoritmo, se crea un programa y será necesaria una computadora para su ejecución.

A continuación, se listan todas las etapas que llevan a la solución de un determinado problema mediante programación.

- Análisis del problema, definición y delimitación.
- Diseño y desarrollo del algoritmo (diagramas de flujo, pseudocódigo, etc.).
- Prueba de escritorio. El algoritmo debe seguirse paso a paso verificando que se realicen todas las instrucciones necesarias para alcanzar el objetivo.
- Codificación. Selección del lenguaje de programación. Escritura del algoritmo utilizando la sintaxis y estructura gramatical del lenguaje seleccionado.
- Compilación. Transformación del lenguaje de programación en lenguaje de máquina.
- Depuración (debug). Proceso de detección y eliminación de los errores de programación.
- Evaluación de resultados. Se debe ejecutar (correr) el programa utilizando datos de entrada y resultados conocidos para verificar que se esté ejecutando el algoritmo adecuadamente ya que es posible que no existan errores de programación (sintaxis) pero los resultados finales no sean los esperados. [11]

Existen diferentes técnicas a través de las cuales se puede ocultar información en un medio digital, con el fin de ser enviado a un receptor, y que el hecho pase inadvertido para terceros. Los principales medios para implementar esteganografía son los de tipo multimedia (video, imagen, sonido). Esta técnica puede implementarse desarrollando un programa de computación en cualquier lenguaje de programación, sin embargo, utilizando el Software matemático Matlab, mediante su lenguaje de programación, se obtienen algunas ventajas respecto al resto. Una de estas ventajas es la forma en que Matlab opera con las imágenes (matricialmente). En este trabajo se expone la implementación de la técnica esteganográfica de Sustitución LSB 1 bit desarrollado en Matlab, haciendo uso de herramientas disponibles para el procesamiento de imágenes. [12]

La popularidad de Internet ofrece una gran comodidad a la transmisión de una gran cantidad de datos a través de las Redes. Sin embargo, también aumenta el riesgo de acceso y manipulación no autorizada de contenido mientras ocurre la transmisión de datos. Para transmitir los datos en las redes de Internet, se deben proporcionar algunos mecanismos para proteger datos importantes contra la interceptación ilegal. Se han propuesto varias técnicas para proporcionar el medio ambiente para la transmisión de datos importantes. Una de las técnicas más importantes es por los datos cifrado [13], en el que los datos están protegidos de acceso ilícito utilizando un algoritmo de cifrado particular.

Los datos son transformados por el algoritmo de cifrado en un texto cifrado que parece una corriente de códigos, y luego se envían al receptor a través de su red. Al recibir el texto cifrado, el receptor puede descifrarlo utilizando una clave de criptografía obtenida del transmisor para recuperar su forma original. El objetivo de la ocultación de datos es que sean inaudibles o invisibles ante los sentidos humanos. De tal manera que el atacante no notara la existencia de datos incrustados, aunque se escuchen muy duro u observen cuidadosamente en los medios en los cuales se ocultan los datos.

Un Portador imagen puede ser utilizado para ocultar, a la vista de un atacante, cualquier mensaje u objeto software (archivo), codificándolo como sutiles cambios en los colores de los píxeles (sus componentes RGB) que no deben ser percibidos por el ojo humano; de tal forma que el Portador que contiene el mensaje, o estegoportador, pueda ser transmitido, sin que sea detectado el hecho, para luego aplicar el proceso inverso (decodificación) de modo que el receptor pueda recuperar y disponer del Mensaje enviado. En general, cualquier archivo multimedia contiene áreas de datos poco significativas. Esas aéreas se pueden sustituir por otros datos, realizando cambios que son imperceptibles a la claridad visual (o auditiva) humana. Esto permite encubrir información de interés dentro de un archivo Portador, haciendo que el mismo parezca igual al original. La teoría indica que con la sustitución del bit menos significativo el ojo y oído, según el portador, no serán capaces de detectar los cambios. En general los métodos de sustitución, y en particular LSB, es de los más efectivos si los archivos portadores tienen mucho ruido, en el sentido de que las imágenes tengan bruscos cambios de color entre pixeles adyacentes, siendo lo opuesto a tener imágenes con amplias áreas de color uniforme. Una característica importante, es que el método de sustitución LSB no incrementa el tamaño del archivo portador, y dependiendo del porte del Mensaje a ocultar, puede causar cambios significativos en el espectro de color del portador respecto a la versión original, lo que puede no ser perceptible, pero si detectable bajo análisis espectral y/o estadístico.

2.7.1 Forma habitual de implementar LSB 1 bit

Para entender mejor la forma general en la cual se implementa la técnica base de sustitución 1 bit, (observar la Ilustración 21). En ésta, se aprecia que de cada byte del mensaje se extrae cada bit, y éste a su vez es insertado en cada bit menos significativo de cada componente de color RGB, o en cada byte, que conforma a cada del pixel de la imagen portadora. En la ilustración podemos ver un ejemplo el cual hace referencia a pixeles de una imagen portadora BMP de 24 bits.

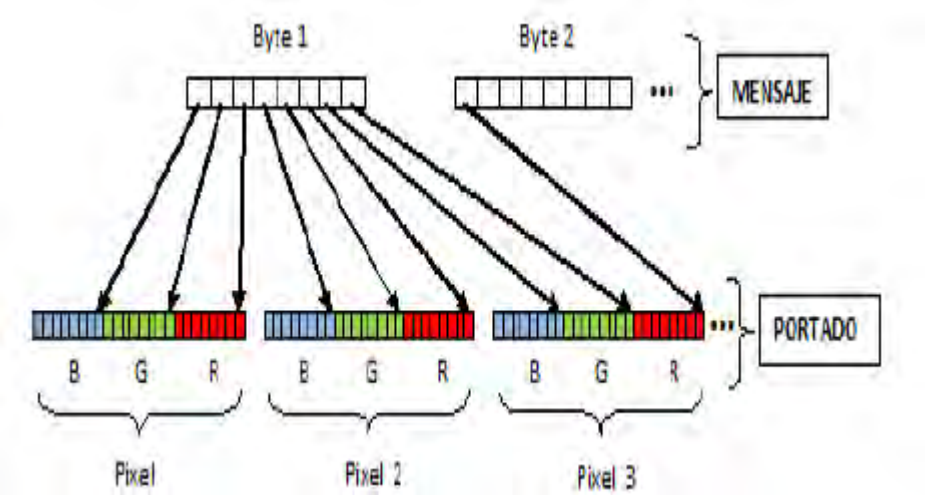


Ilustración 21 Implementación LSB 1 bit

Es importante tener en cuenta que cuando se opera sobre una imagen desde su archivo binario, el orden de los bytes que representa a las componentes R, G y B de cada pixel, presentan el orden B, G y R. ese orden es el que se muestra en la ilustración tratando de la descripción del esquema de implementación de la técnica. Corresponde con el orden que representa cada una de las componentes del color cuando se lee una imagen byte a byte desde su almacenamiento en disco. Siendo también el mismo orden de canales (B, G y R) en el que se trabaja cuando se usa la imagen desde un programa escrito en cualquiera de los lenguajes de programación.

2.8 Técnica de aleatorización

Los números aleatorios son observaciones independientes de variables aleatorias distribuidas uniformemente entre 0 y 1. Observaciones de variables aleatorias no uniformes se obtienen generalmente a través de transformaciones de muestras de números aleatorios uniformes usando semillas.

La semilla es el estado inicial del generador de números, que permite que la secuencia generada por la función sea distinta cada vez que se genere.

2.8.1 Propiedades

Generalmente se quiere que los números cumplan con ciertas propiedades para considerarlos aleatorios.

- Deben ser estadísticamente aleatorios, lo que quiere decir que su distribución es uniforme (que todos los elementos tienen la misma probabilidad de aparecer).
- Impredecibles, es decir que no se pueda predecir el siguiente número de la secuencia luego de observar anteriores números.
- Deben tener periodos largos, ya que como estamos utilizando el modulo, los números eventualmente se van a repetir. [15]
- Eficiencia: Se desea que el generador sea eficiente, para disminuir la carga en algoritmos que podrían ser paralelos.

2.8.2 Pruebas

Existen algunas pruebas estadísticas que se pueden aplicar a los generadores para probar que tan uniforme es su distribución. Estas pruebas pueden ser cualitativas o cuantitativas. Algunos ejemplos:

- Cuantitativas:
  - Pruebas x^2 : Dividen el intervalo de numeros y calculan el esperado de la distribución y lo comparan con el generado para ver si está bien distribuido.
  - Lagged Correlation: Estas pruebas comprueban si es que existe una relación obvia entre dos números de la secuencia.
- Cualitativas:

  - Analisis Visual de la Muestra (Scatter Plots): Se genera un gráfico con la secuencia generada y se busca patrones en la imagen, lo que indicaría una distribución pobre.

  - Pruebas Diehard: Es una batería de pruebas estadísticas que prueban la calidad de los números generados.

2.8.3 Algoritmos

Existen diversos algoritmos con los cuales se puede realizar la secuencia de aleatorización. A continuación de mencionaran algunos de estos algoritmos.

- XOR-shift

Es un algoritmo muy rápido ya que se basa en hacer la secuencia haciendo corrimientos de bits y funciones XOR (” o” exclusivo), estas funciones a nivel de procesador son muy rápidas y

no gastan mucho espacio. Para la semilla podemos usar cualquier número y le hacemos luego los corrimientos de bits y funciones XOR necesarias para crear el patrón.

- Blum Blum Shub

Es un algoritmo un poco más lento que el XOR-Shift u otros ya que este usa exponenciaciones modulares pero este es más seguro, fue creado por, este algoritmo genera bits con la formula $x_i = x_{i-1}^2 mod\ N$ sacando el bit menos significativo del x_i. Este algoritmo es más seguro ya que la variable *N* es grande y difícil de factorizar.

- Mersenne Twister

El Mersenne Twister es el algoritmo de generación de números pseudo-aleatorios más utilizado. Su nombre proviene del hecho que su periodo es un primo Mersenne (primos de la forma $M_p = 2^p - 1$). Fue desarrollado en 1997 por Makoto Matsumoto y Takuji Nishimura. Es el generador de aleatorios utilizado por varios lenguajes de programación incluyendo R, Python, Ruby, Pascal, Matlab, GNU, y C++ desde su versión C++11.

- Dual Elliptic Curve (Dual EC DRBG)

El Dual EC DRBG, es el algoritmo presentado por la NIST (National Institute of Standard Technology) para la generación de bits aleatorios. Este algoritmo requiere unos ciertos valores especificados, los cuales son la curva elíptica, y los dos puntos P y Q. La NSA requería que todos los que querían utilizar este algoritmo de forma aprobada utilicen los siguientes valores.

2.9 Mersenne Twister MT199937

En el presente trabajo se utilizará el algoritmo Mersenne Twister debido a que una de sus características que es la implantación en Matlab, además que es estable ya que las técnicas que utiliza para generar la aleatorización son más inciertas. El nombre proviene del hecho de que la longitud del periodo corresponde a un número primo de Mersenne. Existen al menos dos variantes de este algoritmo, con la única diferencia entre ellos por el tamaño de primos de Mersenne utilizados. El más reciente y más utilizado es el Mersenne Twister Tm19937, con un tamaño de palabra de 32 bits. Existe otra variante con palabras de 64 bits, el MT19937-64, la cual genera otra secuencia.

A continuación, se describirá la forma en que opera este algoritmo.

2.9.1 Descripción

Una computadora es una máquina determinista con un número finito de estados. El problema es cómo generar una sucesión infinita de números aleatorios, con una distribución de probabilidad casi nula, de manera algorítmica.

A lo más que podemos aspirar es a generar una secuencia de números aleatorios, es decir, tales que se comportan para el programa en el que son utilizados como una secuencia de números aleatorios (es decir, no es posible detectar correlaciones estadísticas entre ellos) [16].

Los generadores en base al Método Congruencial Lineal son muy utilizados, pero muchos de ellos tienen un periodo mucho más corto que el que uno desearía o necesita.

El método Mersenne Twister utiliza N celdas para generar los números aleatorios, bajo la siguiente recurrencia:

$$x_{k+n} = x_{k+m} \oplus \left(x_k^u \middle| x_{k+1}^l\right) A, \qquad (k = 0,1, \ldots.)\ donde:$$

- X: entero de w bits
- $\oplus$: XOR
- |: concatenación de cadenas de bits
- x^l:fragmento de bits de X; u son los bits más significativos y l los menos significativos.
- A: matriz de $w\ x\ w$
- n: grado de recurrencias con $1 \leq m \leq n$

El Mersenne twister es un generador de números aleatorios desarrollado en 1997 por Makoto Matsumoto y Takuji Nishimura reputado por su calidad.

Su nombre proviene del hecho de que la longitud del periodo corresponde a un número primo de Mersenne. El más reciente y más utilizado es el Mersenne Twister MT19937, con un tamaño de palabra de 32-bit. Existe otra variante con palabras de 64 bits, el MT19937-64, la cual genera otra secuencia. Este método trabaja a más bajo nivel, dado que utiliza operaciones lógicas en cadenas de bits, y tiene un periodo de repetición de $2^{19937} - 1$.

2.9.2 Definición, propiedades y aplicaciones de los números primos de Mersenne.

Se considera que un número es de Mersenne si es un número de la forma:

$$M_n = 2^n - 1$$

Adicionalmente se dice que un número de Mersenne es primo de Mersenne si es un número primo de la forma:

$$M_{p=2^p-1}\ con\ p\ primo.$$

El estudio de este tipo de números supone un área de especial interés por varios motivos, entre ellos las relaciones que se producen entre los primos de Mersenne y otras categorías de números útiles en distintos contextos. Los números de Mersenne son de base binaria, esta propiedad resulta interesante, sobre todo al realizar operaciones con computadoras, ya que éstos trabajan en base binaria, puesto que ciertos cálculos se simplifican en extremo. Estos números además guardan una estrecha relación con los números perfectos, dado el hecho de que si M es un número primo de Mersenne entonces $M * \frac{M+1}{2}$ es un número perfecto. Un número perfecto es un número natural que es igual a la suma de sus divisores propios positivos sin incluirse él mismo.

Mencionaremos también los números dobles de Mersenne, que son primos de la forma:

$$M_{Mp} = 2^{(2^p)-1}\ con\ p\ primo$$

Todos estos números primos son de gran importancia, especialmente en aplicaciones criptográficas, ya que los números primos más grandes que se conocen son números primos de Mersenne. Debido a sus propiedades características, algunas de las cuales ya hemos mencionado, es posible acelerar procesos asociados a la criptografía de clave pública y a los procesos de criptografía basados en curvas elípticas.

2.9.3 Ejemplo de implementación

A continuación, se va a presentar un ejemplo sobre el código para la implementación de este algoritmo.

El Mersenne twister consta de tres partes que nos permiten generar los números aleatorios. Primero declaramos un vector estático de 624 números, en este caso variables ZZ de la librería

NTL para trabajar números grandes, y un índice que nos va a permitir navegar el vector. Además, tenemos la inicialización del vector a partir de una semilla como se muestra en la ilustración.

```
/*--------------------Mersenne Twister--------------------*/
// Declaraciones
ZZ mersenne_state[624];
int m_index;
//Inicializador del vector
void MT_init(ZZ seed){
    m_index = 0;
    mersenne_state[0] = seed*seed;
    for (int i = 1; i < 624; ++i){
        ZZ x;
        x = mersenne_state[i-1]<<30;
        x = mod(x,conv<ZZ>(MAX_XOR));
        ZZ tmp = mersenne_state[i-1];
        ZZ ans;
        ans = x^tmp;
        string mult = "18124332531873678163";
        ans *= conv<ZZ>(str_to_ZZ( mult));
        mersenne_state[i] = abs(ans);

    }

}
```

Ilustración 22 Declaraciones e inicialización del vector.

El vector se inicializa con el valor de la semilla al cuadrado y el índice con 0. Luego llenamos el vector en las posiciones 1 a la 623 haciendo una serie de, xor y multiplicaciones. Luego vamos a necesitar una función que regenere el vector cuando termine de generar los números aleatorios. [17]

```
//Regenera el vector
void mt_regenerate() {
    for (int i = 0; i < 624; ++i)
    {
        ZZ y = (mersenne_state[i] & 0x80000000)+(mersenne_state[(i
            +1)%624] & 0x7fffffff) ;
        mersenne_state[i] = mersenne_state[(i+397)%624]^(y>>1);
         if( mod(y,conv<ZZ>(2)) != 0 ){
             mersenne_state[i] = mersenne_state[i]^(2567483615);
         }
    }
}
```

Ilustración 23 Muestra de regeneración del vector.

Esta función se va a mandar a llamar una vez cada 624 pedidas de números debido a que es el valor del vector estadístico generado inicialmente. Como vamos a ver a continuación en la función que es la que usamos para pedir números.

```
1  //Retorna un numero aleatorio, regenera el vector cada 624 llamadas
2  ZZ mt_rand() {
3       if (m_index == 0){
4           mt_regenerate();
5       }
6
7       ZZ y = mersenne_state[m_index];
8       y = y^(y>>11);
9       y = y^(y<<7 & (conv<ZZ>(2636928640)));
10      y = y^(y<<15 & (conv<ZZ>(4022730752)));
11      y = y^(y>>18);
12
13      m_index++;
14      m_index = m_index%624;
15      return y;
16 }
```

Ilustración 24 Regeneración del vector cada 624 pedidas de números

En esta función igualmente se hace uso del XOR binario y los corrimientos para que los números sean aún más distintos. La forma de utilizar el Mersenne Twister que hemos implementado es la siguiente:

```
1  int main(){
2       MT_init(conv<ZZ>(time(NULL)));
3       for(int i=0; i<10 ; i++){
4         cout<<mt_rand()<<endl;
5       }
6  }
```

Ilustración 25 Implementación de Mersenne Twister

Este código imprimirá 10 números aleatorios y está utilizando el tiempo del sistema como semilla.

Capítulo 3 Desarrollo

Una vez revisado las partes más importantes que son utilizadas para implementar algoritmos de esteganografía mediante el algoritmo LSB, a continuación, se describe el desarrollo del algoritmo alojado en el transmisor (Tx) y en el receptor (Rx).

3.1 Algoritmo en Matlab

La implementación en Matlab del algoritmo LSB fue en una imagen como archivo digital la cual tiene las siguientes características:

- Escala de grises
- Tipo de archivo: PNG
- Ancho: 241 pixeles
- Largo: 239 pixeles

El tamaño del mensaje a insertar depende de las características de la imagen especialmente el ancho y largo. Tomando en cuenta las características de la imagen que se está utilizando tenemos un total 241x 239= 57599 bytes de los cuales se utilizara un solo bit de cada pixel, por lo tanto, tendremos un total de 57599 bits para ocultar información teniendo la posibilidad de ocultar un mensaje de 7199 caracteres.

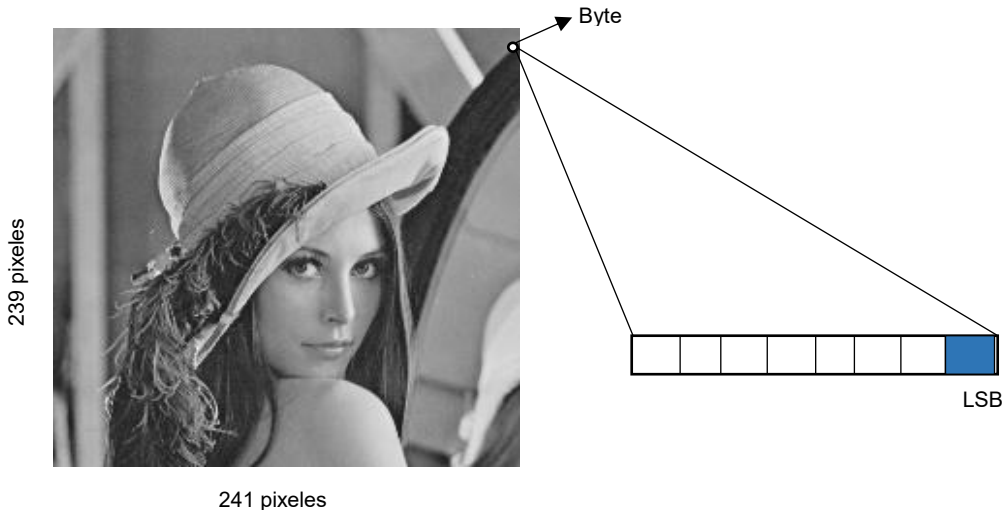


Ilustración 26 Características de imagen que se está utilizando

A continuación, se describe el proceso de realización del algoritmo LSB. En la Ilustración 27, se muestra un diagrama a bloques, el cual es llevado a cabo para realizar la versión final del algoritmo en el lenguaje de programación Matlab, mostrando el proceso realizado en el transmisor Tx.

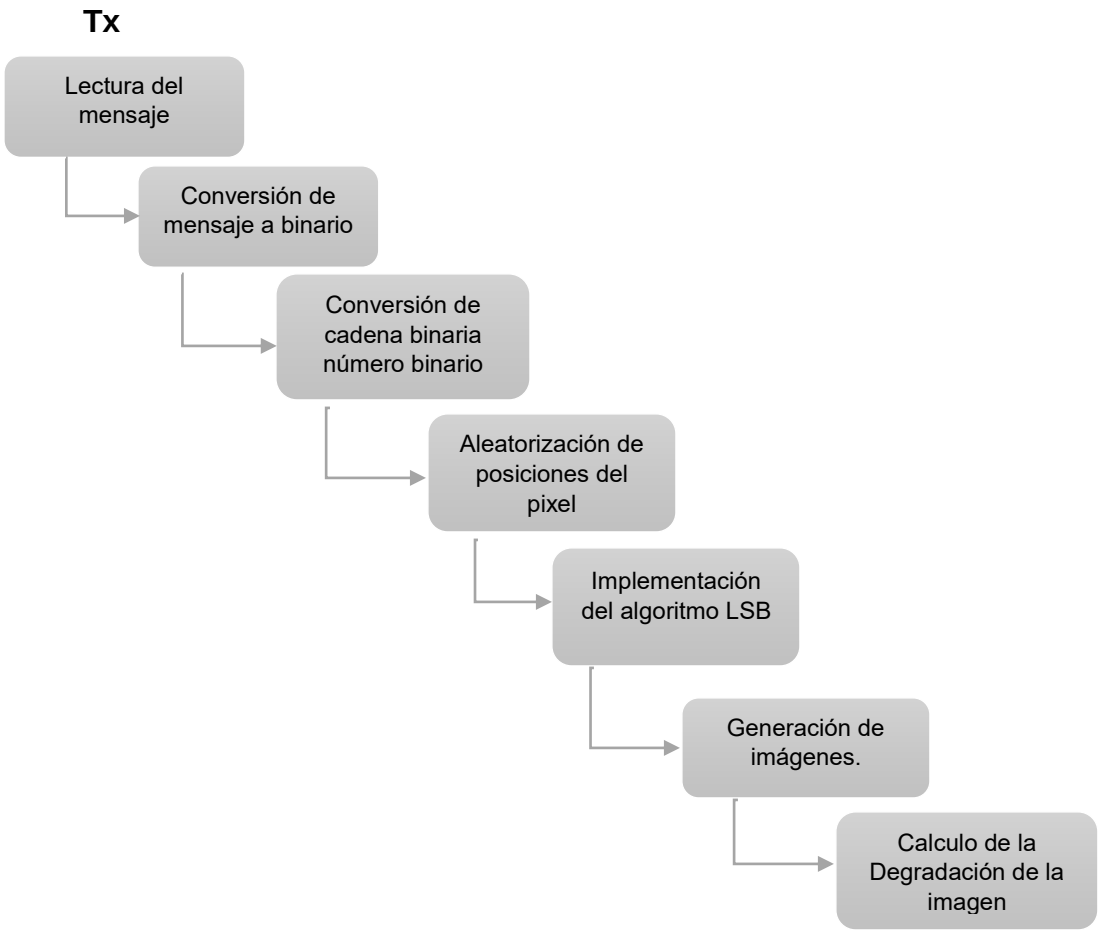


Ilustración 27 Proceso de transmisión del mensaje

De igual manera se muestra el proceso de transmisión de un mensaje, iniciando con la lectura, la conversión, y finalizando con la inserción de bits a la imagen.

La Ilustración 28 muestra un diagrama a bloques sobre el proceso en el lado receptor Rx del mensaje, el cual consta de 4 bloques.

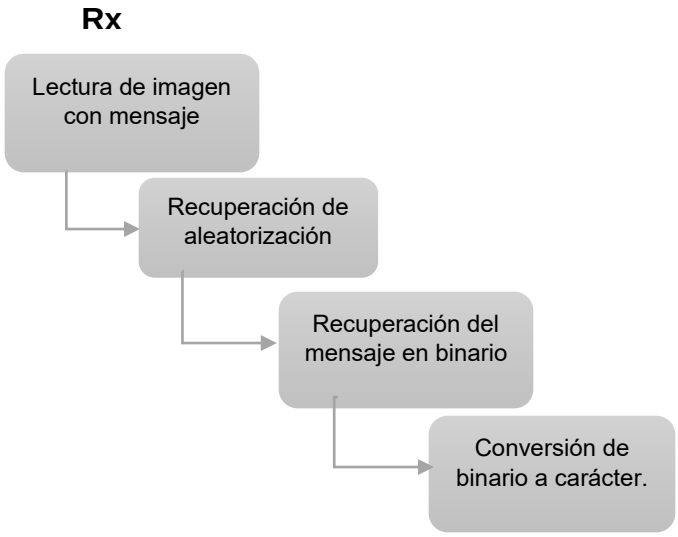


Ilustración 28 Proceso de recepción del mensaje

3.1.1 Transmisor Tx

La Ilustración 29 muestra el bloque llamado lectura del mensaje que describe la inicialización del algoritmo implementado en Matlab inicializando con borrar todas las entradas y salidas de la ventana de comandos, después se asigna el mensaje el cual es leído como una cadena y se calcula la longitud de dicho mensaje.

```
1 -    clear all;
2 -    clc;
3 -    c = imread('lenna_gris.png');
4 -       message = 'ABCDEFGHIJKLMNOPQRZ12345678910ABCDEFGHIJKLMNOPQRZ12345678910!"#$%&/()()()()()ABCDEFGHIJKLMNOPQRZ1234';
5       %   message = '- -Esta es una prueba de esteganografia a traves de substitucion de bit menos significativo o LSB- -';
6 -       message = strtrim(message);
7 -       m = length(message) * 8;
```

Ilustración 29 Bloque: Lectura del mensaje

La Ilustración 30 muestra el bloque llamado conversión del mensaje a binario, en donde aún se contiene el mensaje en la línea 8. Así, se hace la conversión de dicha cadena de texto a código ASCII con el fin de representar caracteres alfanuméricos. Para continuar con el proceso, se procede a realizar una conversión a lenguaje binario (en la línea 9) que es la representación de

1s y 0s. En línea 10, la cadena binaria se genera en un vector fila, y posteriormente se calcula la longitud de este nuevo vector binario.

```
8 -         AsciiCode = uint8(message);
9 -         binaryString = transpose(dec2bin(AsciiCode,8));
10 -        binaryString = binaryString(:);
11 -        N = length(binaryString);
```

Ilustración 30 Bloque: Conversión del mensaje a binario

Para ejemplificar el proceso de inserción de un mensaje, supongamos que se desea enviar un carácter A. En alfanumérico la letra A es representada en código ASCII con el número 65 por lo que este número en representación binaria corresponde de la siguiente manera: 01000001.

La ilustración 31 muestra el bloque llamado conversión de cadena binaria a numero binario, de la línea 12 a la línea 19 se realiza el proceso para cambiar de cadena binaria a número binario. Para lo cual se crea un nuevo vector, el cual será llenado en base a la condición del ciclo **`for`**.

```
12 -        b = zeros(N,1); %b is a vector of bits
13 -        for k = 1:N
14 -            if(binaryString(k) == '1')
15 -                b(k) = 1;
16 -            else
17 -                b(k) = 0;
18 -            end
19 -        end
```

Ilustración 31 Bloque: conversión de cadena binaria a número binario

La Ilustración 32 muestra el bloque llamado aleatorización de posiciones del pixel en el cual de la línea 20 a la línea 26 se realiza el procedimiento de la aleatorización mediante el método Twister. En donde se nombra otra variable la cual va a contener la imagen original, para posteriormente calcular la anchura y altura de la imagen. En la línea 24 se indica la inicialización de la semilla, que para este ejemplo inicializa en 1 y con el comando `rng(k,'twister');` siembra el generador de números aleatorios unirformes usando una semilla de números enteros no negativos para que la función `randi` produzca una secuencia predecible de números.

```
20 -        s = c;
21 -        height = size(c,1);
22 -        width = size(c,2);
23 -        k = 1;
24 -        rng(k,'twister'); % aleatorizaciön de la posiciön
25 -        pos_h = randi(height,1,N+1);
26 -        pos_w = randi(width,1,N+1);
```

Ilustración 32 Bloque: Aleatorización de posiciones del pixel

En la Ilustración 33 se muestra el bloque llamado implementación del algoritmo LSB en sí. Como puede observarse, con el ciclo for se realiza el proceso de lectura y modificación de bits, tomando valores de 1 hasta N en donde N representa la longitud del mensaje en un vector de bits. Posteriormente, se leen las posiciones aleatorias que se generaron anteriormente, (en la línea 31), se realiza la lectura del ultimo bit de la imagen para posteriormente tomar en cuenta la longitud del mensaje y hacer una comparación entre ésta y b(k) en la posición k. Esta posición depende del número binario correspondiente generado en el bloque conversión. La inserción de los bits es realizada con el código descrito de las líneas 35 a 38.

```
28 -    for l = 1:N
29 -            i = pos_h(l);
30 -            j = pos_w(l);
31 -             LSB = mod((c(i,j)), 2);
32 -            if (k>m || LSB == b(k))
33 -                s(i,j) = c(i,j);
34 -            else
35 -                if (LSB==1)
36 -                    s(i,j) = c(i,j) - 1;
37 -                else
38 -                    s(i,j) = c(i,j) + 1;
39 -                end
40 -            end
41 -            k = k + 1;
42 -    end
```

Ilustración 33 Bloque: Implementación del algoritmo LSB

En la siguiente ilustración se muestra el bloque llamado generación de imágenes, en el cual de la línea 42 a la línea 46 se crean y muestran 2 figuras de las cuales la primera es la original y la segunda es el resultado de la inserción de información a través del algoritmo LSB. Posteriormente, en la línea 47 se escriben los datos de la matriz imagen s en una imagen llamada '111.png' y de esta forma se genera un nuevo archivo digital.

```
43 -        figure(1)
44 -        imshow(c)
45 -        figure(2)
46 -        imshow(s)
47 -        imwrite(s, '111.png');
```

Ilustración 34 Bloque: generación de imágenes

En ilustración 35 muestra el bloque llamado cálculo de la degradación de la imagen que implementa una métrica de la señal a ruido. La relación señal a ruido (SNR) define la proporción existente entre la potencia de la señal que se transmite y la potencia del ruido indeseado, en este caso la imagen original y la imagen que se genera después de la inserción de bits. La medición es en dB y es una medida de cualquier cantidad en relación a un nivel conocido, es una unidad logarítmica que nos permite representar números muy pequeños o muy grandes.

```
48          %% SNR computation
49 -        Power_c_image     = 0;
50 -        Power_error_image = 0;
51 -           for i = 1 : height
52 -             for j = 1 : width
53 -                 dum_c     = double (c(i,j));
54 -                 dum_s     = double (s(i,j));
55 -                 dum_noisy = dum_c-dum_s;
56 -                 Power_c_image     = Power_c_image + dum_c.^2;
57 -                 Power_error_image = Power_error_image + (dum_noisy.^2);
58 -             end
59 -           end
60
61 -           SNR_db = 10 * log10(Power_c_image/Power_error_image)
```

Ilustración 35 Bloque: Calculo de la degradación de la imagen

3.1.2 Receptor Rx

La ilustración 36 muestra el bloque llamado Lectura de imagen con mensaje. En donde se inicia leyendo la imagen que se generó en el bloque generación de imágenes del Transmisor Rx, se realiza el cálculo de las dimensiones de la imagen, y se asigna a una variable la longitud del mensaje (en la línea 5).

```
1 -         s = imread('lll.png');
2 -         height = size(s,1);
3 -         width = size(s,2);
4           %For this example the max size is 100 bytes, or 800 bits, (bytes * = bits
5 -         m = N;
6 -         k = 1;
```

Ilustración 36 Bloque: Lectura de imagen con mensaje

La ilustración 37 muestra el bloque llamado Recuperación de la Aleatorización en donde se reinicia el generador de números uniformes aleatorios al del transmisor y de esta manera se recuperan las posiciones de los pixeles en donde se realizó la modificación de los bits haciendo la inserción del LSB.

```
7 -         rng(l,'twister');
8 -         pos_h = randi(height,l,N+l);
9 -         pos_w = randi(width,l,N+l);
```

Ilustración 37 Bloque: Recuperación de Aleatorización

La siguiente 38 muestra el bloque llamado Recuperación del Mensaje en Binario, en donde en base en las posiciones que se generaron se recuperan los valores binarios introducidos desde 1 hasta N; en donde N corresponde a la longitud del vector unitario del mensaje.

```
10 -    for l = 1:N
11 -            i = pos_h(l);
12 -            j = pos_w(l);
13
14 -            if (k <= m)
15 -                br(k) = mod(double(s(i,j)),2);
16 -                k = k + 1;
17 -            end
18 -     end
```

Ilustración 38 Bloque: Recuperación del Mensaje en Binario

La ilustración 39 muestra el bloque llamado Conversión de Binario a Carácter en donde se hace la conversión para lograr recuperar el mensaje que se ocultó en la imagen, se escriben los valores correspondientes a el octeto (en la línea 20), se genera un vector binario. Posteriormente, se realiza la conversión por lo que la función reshape ordena el vector binario dividido en ochos para formar los octetos del byte y lo almacena en una matriz, finalmente la función char en la línea 26 convierte esta matriz en cadena de caracteres.

```
19 -        binaryVector = br;
20 -         binValues = [ 128 64 32 16 8 4 2 1 ];
21 -         binaryVector = binaryVector(:);
22 -         if mod(length(binaryVector),8) ~= 0
23 -             error('Length of binary vector must be a multiple of 8.');
24 -         end
25 -         binMatrix = reshape(binaryVector,8,100);
26 -         textString = char(binValues*binMatrix);
27 -         disp(textString);
28 -    bt = b';
29 -    Errores = sum (br-bt)
```

Ilustración 39 Bloque: Conversión de Binario a Carácter.

Capítulo 4 Resultados

Este proyecto de tesis se realizó en Matlab con la versión R2015a y se utilizó una laptop HP con las siguientes características:

- Procesador Intel(R) Celeron(R) 1.60 GHz
- Memoria RAM de 4.00 GB
- Sistema operativo Windows 10 de 64 bits

Se ha desarrollado en Matlab una función denominada LSB, a través de un archivo (.m), que implementa el algoritmo de la técnica de Sustitución LSB 1 bit, operando a la imagen contenedora del mensaje oculto como una matriz de la imagen. El código desarrollado el algoritmo de sustitución LSB 1 bit implementado, generando como resultado la presentación gráfica por pantalla de la imagen original (portadora), así como también a la imagen en donde se realizó la inserción de los bits (estegoportadora).

Se presenta el resultado generado por el algoritmo implementado en la función LSB1( ). Se elige como Portador a la imagen "lenna_gris.png" de 241x239 pixeles como se muestra en la ilustración 40.

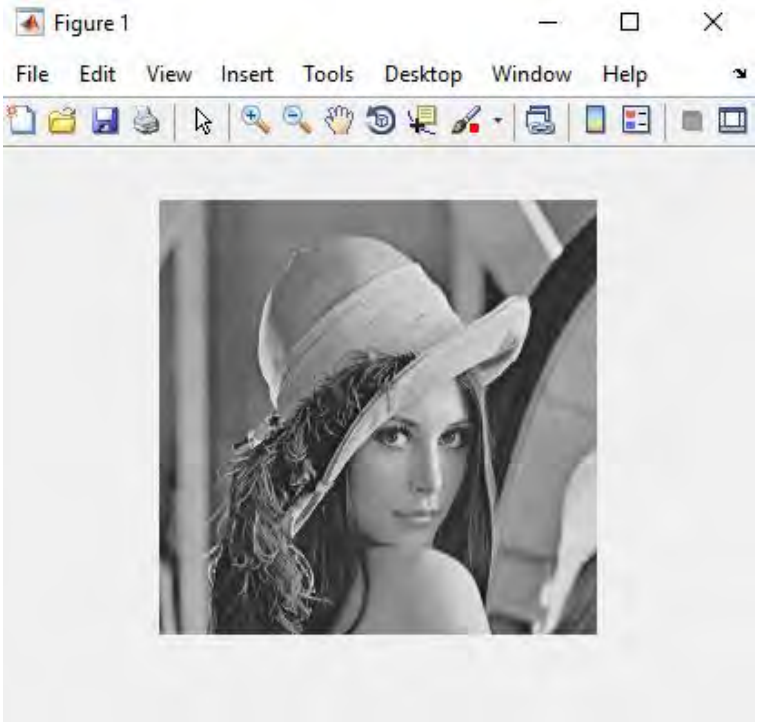


Ilustración 40 lenna_gris portadora

Como mensaje la cadena de texto:
'ABCDEFGHIJKLMNOPQRZ12345678910ABCDEFGHIJKLMNOPQRZ12345678910!"#$%&/()()()()()ABCDEFGHIJKLMNOPQRZ1234'
la siguiente cadena fue elegida con el fin de mostrar la posibilidad de representar números, letras y caracteres. Con un total de 100 caracteres, una representación de 8 bits por caracteres; teniendo un total de 800 bits para ocultar en la imagen portadora.

Otra segunda prueba se utilizó la siente cadena de texto:

- -Esta es una prueba de esteganografia a traves de substitucion de bit menos significativo o LSB- -
Con la finalidad de demostrar que los espacios son tomas como caracteres en la implementación del LSB.

Teniendo como resultado al archivo imagen "111.png" con las mismas dimensiones de la imagen inicial, es decir, 241x 239 pixeles. Como se muestra en la ilustración 41.

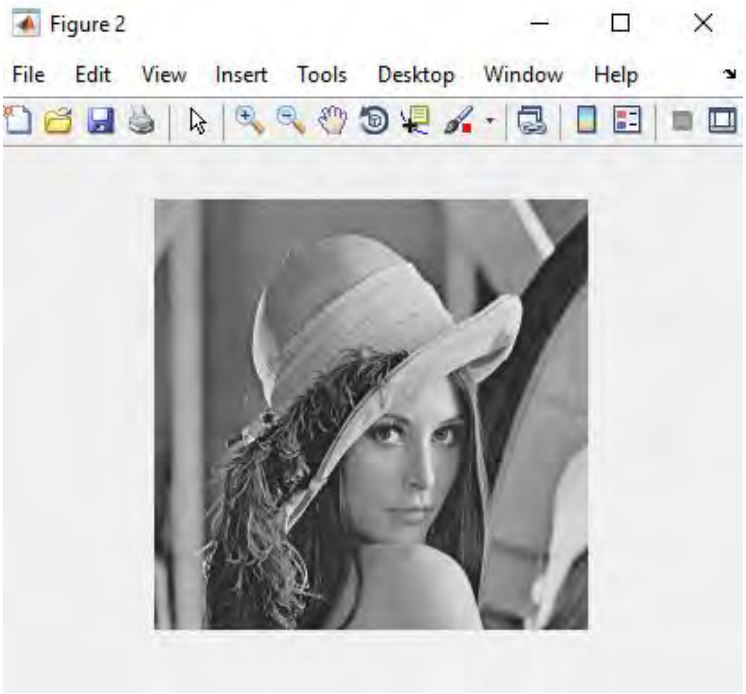
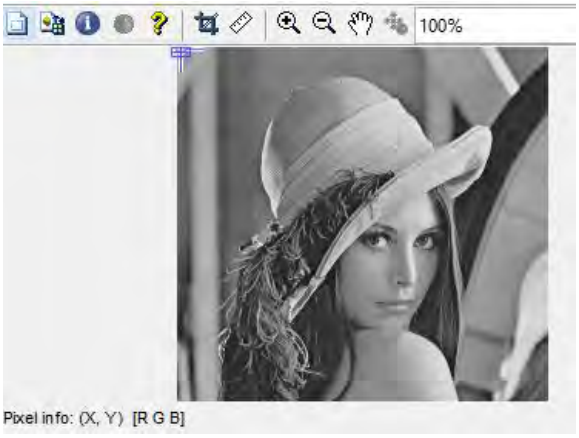


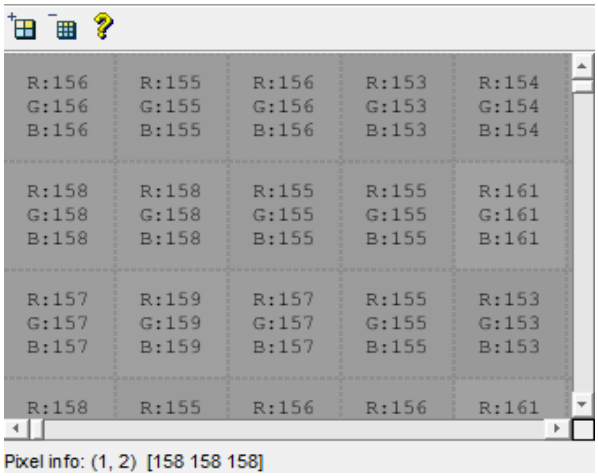
Ilustración 41 Imagen con el algoritmo LSB(111.png)

Matlab cuenta con la función imtool la cual abrirá un visor de imágenes que muestra el nivel de gris de la imagen, tomando una región de pixeles.

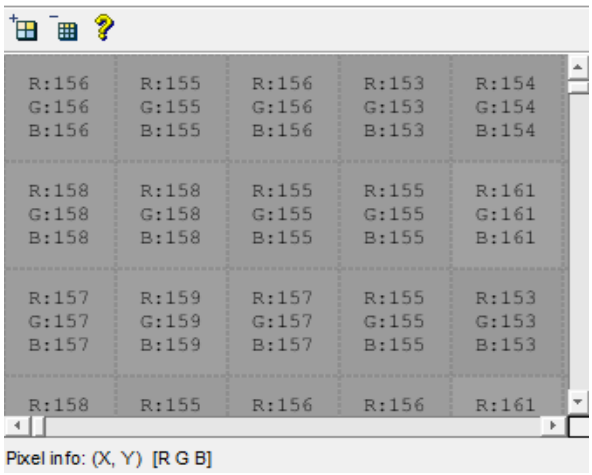
A) Imagen Tool 1 C



B) Imagen Tool 2 S



C) Región de pixeles (Imagen Tool 1)



D) Región de pixeles (Imagen Tool 2)

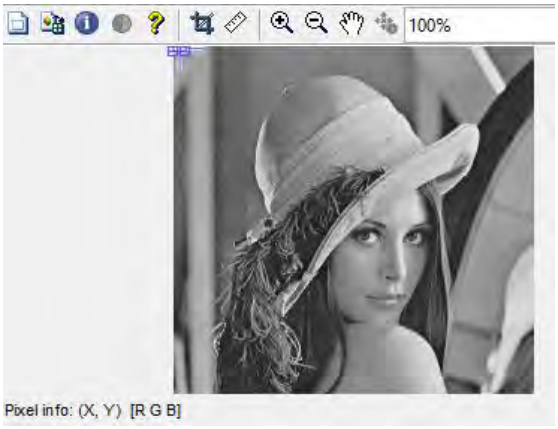
Ilustración 42 Representación de la modificación de los bits

En la ilustración 42(A) muestra la imagen original que fue utilizada para la inserción del mensaje, la ilustración 42(B) es la figura que se genera una vez que se realiza el proceso de inserción del bit menos significativo (LSB), en la ilustración 42(C), (imagen portadora) se muestra una región de pixeles de la figura original, que será comparada con la ilustración 42(D) que es la región en donde ya fueron modificados los bits.

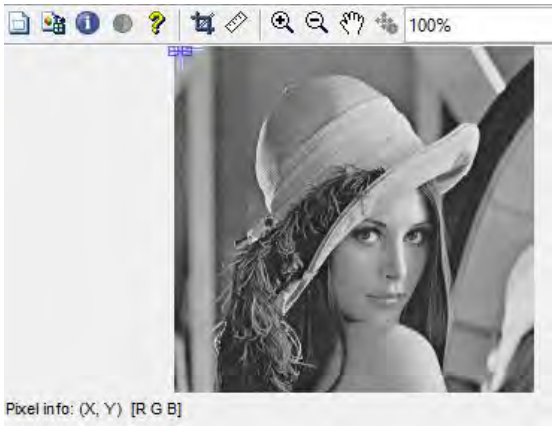
Tomando en cuenta la región de pixeles que fueron modificados, se obtiene como resultado que visiblemente no existe ningún cambio y haciendo la comparación de los valores entre la ilustración 42(C) y la ilustración 42(D) son iguales y no existe ninguna modificación. Esto se debe a que la longitud del mensaje que se inserta (100 caracteres o una representación binaria

de 800 bits) es muy pequeña en comparación a la longitud máxima de la información que se puede ocultar. De la misma forma como se realizó el proceso de aleatorización es muy impredecible deducir en donde fueron modificados los bits.

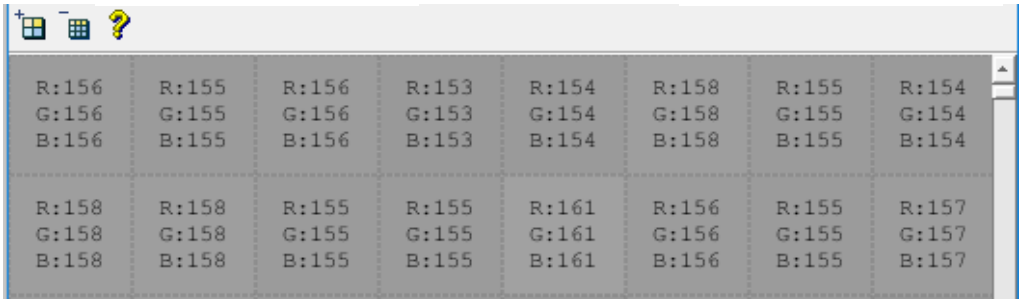
Con el objetivo de probar la funcionalidad del algoritmo LSB se realizaron pruebas sobre este mismo, utilizando el código sin la parte de aleatorización, de tal manera que la inserción se realiza de manera secuencial con el motivo de fácil identificar los cambios de valores de bits que fueron modificados, como se muestra en la ilustración 43.



A) Imagen Tool 1 c

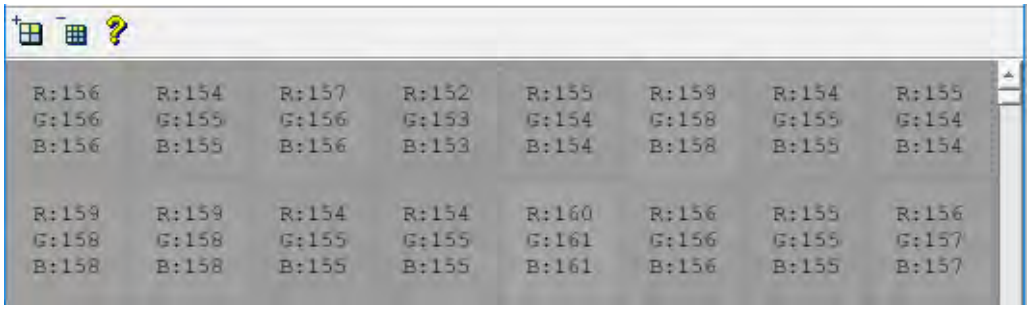


B) Imagen Tool 1 c



R:156 G:156 B:156	R:155 G:155 B:155	R:156 G:156 B:156	R:153 G:153 B:153	R:154 G:154 B:154	R:158 G:158 B:158	R:155 G:155 B:155	R:154 G:154 B:154
R:158 G:158 B:158	R:158 G:158 B:158	R:155 G:155 B:155	R:155 G:155 B:155	R:161 G:161 B:161	R:156 G:156 B:156	R:155 G:155 B:155	R:157 G:157 B:157

C) Región de pixeles (Imagen Tool 1 c)



R:156 G:156 B:156	R:154 G:155 B:155	R:157 G:156 B:156	R:152 G:153 B:153	R:155 G:154 B:154	R:159 G:158 B:158	R:154 G:155 B:155	R:155 G:154 B:154
R:159 G:158 B:158	R:159 G:158 B:158	R:154 G:155 B:155	R:154 G:155 B:155	R:160 G:161 B:161	R:156 G:156 B:156	R:155 G:155 B:155	R:156 G:157 B:157

D) Región de pixeles (Imagen Tool 2 s)

Ilustración 43 Demostración de funcionalidad del algoritmo LSB

Para una mejor explicación sobre la ilustración anterior se tomará una cadena de los primeros 8 bits pertenecientes a el vector binario b(k) correspondiente al ciclo **`for`**, en donde realiza la conversión de cadena binaria a número binario del mensaje a insertar en la imagen (ver Anexo C del código sin aleatorización) para demostrar el proceso de inserción.

Para estas pruebas el mensaje a transmitir es el siguiente:

```
'- -Esta es una prueba de esteganografia a traves de substitucion de bit menos significativo o LSB- -'
```

De tal manera que los primeros 8 bits corresponden a el carácter `'-'` que representado en código ASCII equivale al número 45 y en representación binaria equivale al siguiente número binario: 00101101 (ver ilustración 44).

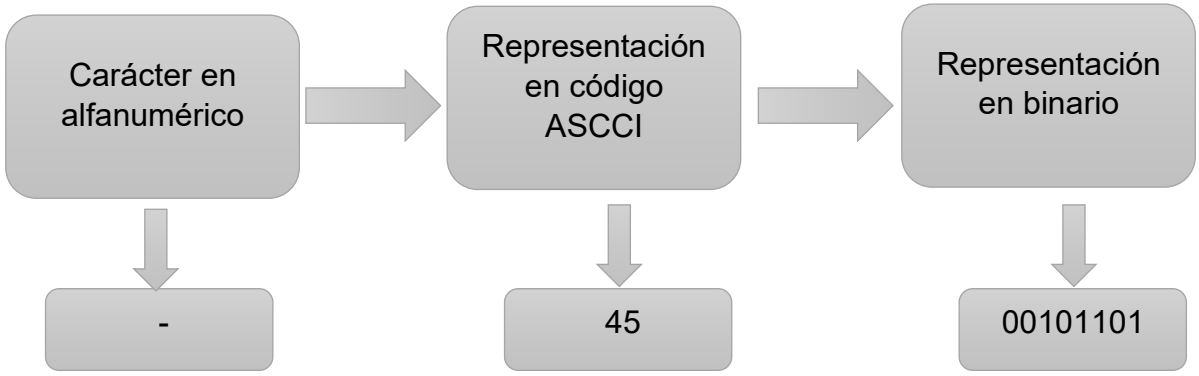


Ilustración 44 Representación de la conversión del carácter '-' a numero binario

En la ilustración 45 se identifican las posiciones de los pixeles que fueron modificados en la imagen c (imagen portadora), tomando los primeros 8 pixeles de la imagen para explicar la representación de los primeros 8 bits de la cadena del mensaje.

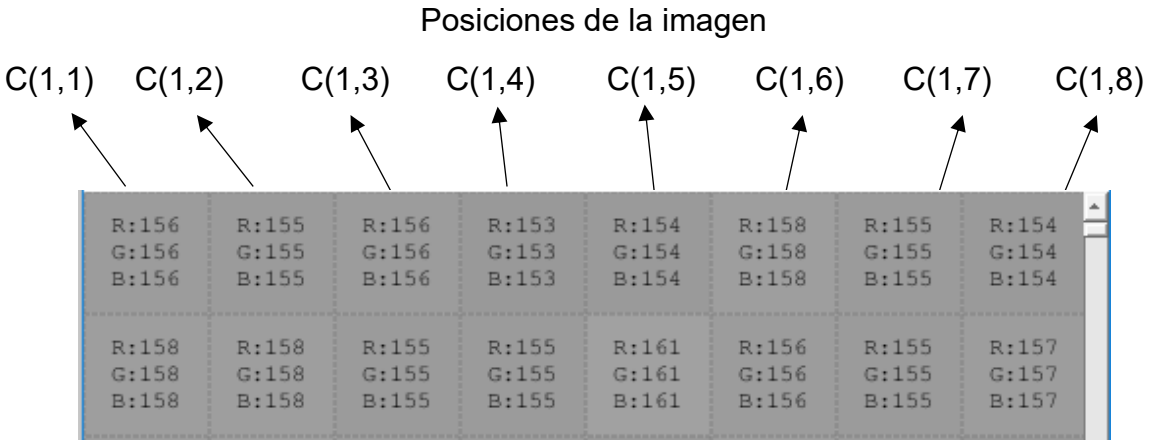


Ilustración 45 Posición sin aleatorización

En escala de grises los valores que pueden tomar van de 0 a 255, en donde el 0 representa al color negro hasta el 255 que corresponde al color blanco.

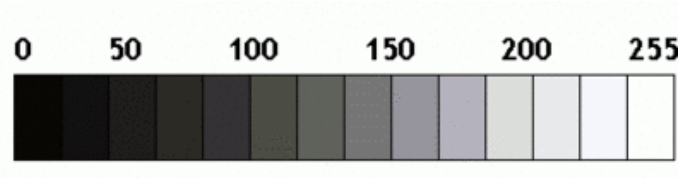


Ilustración 46 Valores del pixel en escala de grises

En la siguiente tabla se explica el proceso en el que se lleva a cabo la inserción de los bits.

Representación de los primeros 8 bits a insertar	Valor del pixel en la posición c(i,j)	Valor en binario de la posición c(i,j) Valor	Resultado del pixel en imagen s(imagen estegoportadora)
0	156	1001110**0**	1001110**0**
0	155	1001101**1**	1001101**0**
1	156	1001110**0**	1001110**1**
0	153	1001100**1**	1001100**0**
1	154	1001101**0**	1001101**1**
1	158	1001111**0**	1001111**1**
0	155	1001101**1**	1001101**0**
1	154	1001101**0**	1001101**1**

Tabla 4 inserción de los primeros 8 bits de la cadena binaria del mensaje

El primer valor de la fila 1 que se desea insertar es **0**, pero el valor del byte en la posición c(1,1) es 156 que en binario equivale al número 1001110**0** y vemos que el ultimo bit es equivalente al valor del bit que se desea insertar por lo tanto se le suma un cero (ver el bloque llamado Implementación del LSB líneas 35-38, del Capítulo 3) de esta forma conserva su valor 1001110**0**.

El segundo valor de la fila 2 que se desea insertar es **0** y el valor del byte en la posición c(1,2) es igual a 155 que en número binario corresponde a 1001101**1**, en este caso el número que se desea insertar es distinto al valor del bit menos significativo al pixel correspondiente. Por lo que

se le resta uno para que el bit menos significativo sea equivalente al bit que se desea insertar, quedando de la siguiente manera 1001110**0**.

Conclusiones

La esteganografía es una técnica en constante evolución, con una larga historia y con capacidad para adaptarse a nuevas tecnologías. Los archivos contenedores del mensaje a enviar no tienen que ser forzosamente imágenes, cualquier medio digital es válido (audio, video, etc.). En cualquier caso, la eficiencia de las nuevas técnicas de esteganografía hace necesario el uso de la esteganografía combinada con criptografía con el fin de alcanzar un nivel de seguridad razonable. La criptografía garantiza la confidencialidad de una conversación, pero no esconde el hecho de que dicha conversación se está manteniendo. Por otra parte, la esteganografía puede ocultar el hecho de que una conversación se mantiene, pero una vez descubierta la interacción, es posible que un atacante conozca el contenido intercambiado. Considerando que descubrir el contenido original fuera difícil, un atacante puede realizar alguna modificación en el transmisor para impedir la comunicación (ataque activo). Conjugando ambas técnicas se alcanza una complementariedad que multiplica la seguridad de un intercambio de mensajes.

El estudio que se realizó sobre la esteganografía permite obtener una visión amplia sobre el panorama actual de la seguridad y se compone una base en la cual se puede seguir desarrollando métodos más especializados. Este estudio, además, me ha permitido establecer una base a partir de la cual se desarrollé un código en Matlab implementando el algoritmo LSB. El estudio de las técnicas de aleatorización permite conocer mejor las debilidades y vulnerabilidades de los algoritmos esteganográficos, que fueron fundamentales para el desarrollo de este trabajo de tesis.

Se propuso un algoritmo con el que se desarrolló este trabajo de tesis, se analizó una técnica para la aleatorización de posiciones de la lectura de los bits que fueron modificados y de esta manera obtener una mayor seguridad en el mensaje que se va a transmitir porque la probabilidad de obtener estas posiciones es prácticamente nula.

Así mismo, se han dado ejemplos y casos en donde la esteganografía puede ser aplicada demostrando las técnicas de funcionamiento a partir de la elaboración de un código en Matlab que realiza la inserción del bit menos significativo (LSB).

Por mencionar uno de los casos donde la esteganografía fue aplicada es en la integridad y autenticación de los datos, ya que esta se utiliza en un juicio en caso de que sea necesario presentar algunas pruebas en fotografías o videos de una cámara de vigilancia. En el caso de las etiquetas números de serie y huellas digitales el proceso consiste en generar un identificador tanto del emisor como el receptor de tal manera que los datos de identificación sean únicos y poder darnos cuenta cuando estos están siendo compartidos a terceras personas. En cualquier de los casos en donde esta se aplica, es necesario la inserción de algunas características de los autores o identificadores para poder probar que esta imagen es real y no fue alterada mediante algún software.

Al no tener el conocimiento necesario sobre el tema para poder desarrollar este trabajo, fue necesario realizar una investigación de los antecedentes, trabajos propuestos hasta ahora, etc. para conocer el impacto de este tema en la actualidad referente a la seguridad de la información. De igual manera, entender la forma en que opera Matlab ya que mi conocimiento de este lenguaje de programación era el mínimo.

Bibliografía

[1] G. G. Paredes, «Introducción a la criptografía,» *Revista digital Universitaria,* vol. 7, p. 17, 2006.

[2] A. G. Konhein, «Computer security and cryptography,» United States ofAmerica, 2007, pp. 1-15.

[3] R. D. V. Velasco, «Criptografía, una necesidad moderna,» vol. 7, 2006.

[4] A. Muños, «Criptored.upm.es,» 2 Enero 2014. [En línea]. Available: http://www.criptored.upm.es/crypt4you/temas/privacidad-proteccion/leccion7/leccion7.html. [Último acceso: 20 Abril 2017].

[5] D. Artz, «Digital Steganography: Hiding Data within Data,» *IEEE Internet computing,* p. 6, 2001.

[6] E. A. M. Checa, JUlio 2014. [En línea]. Available: http://bibdigital.epn.edu.ec/handle/15000/8062. [Último acceso: 3 Julio 2017].

[7] A. M. Balleste, «Esteganografía en contenido multimedia,» 2007.

[8] P. A. Deymonnaz, «Facultad de Ingeniería Buenos Aires,» 12 Marzo 2012. [En línea]. Available: http://materias.fi.uba.ar/7500/TesisDeymonnaz.pdf. [Último acceso: 17 Marzo 2017].

[9] P. A. Deymonnaz, «Análisis de vulnerabilidades esteganográficas en protocolos de comunicación IP y HTTP,» Facultad de Ingeniería, Buenos Aires, 2012.

[10] «Programación en Matlab,» [En línea]. Available: http://www.esi2.us.es/~jaar/Datos/FIA/T9.pdf. [Último acceso: 19 Julio 2017].

[11] I. E. D. I. P. Mores, «Universidad Tecnológica Nacional Facultad Regional Rosario,» 2011. [En línea]. Available: http://www.modeladoeningenieria.edu.ar. [Último acceso: 19 Julio 2017].

[12] A. M. M. F. R. r. Ing. Maria A Gerardino Garcia, «Sistema esteganográfico,» *Revista digitan Universitaria,* vol. 9, nº 4, p. 13, 2008.

[13] C.-F. L. J. C. L. Ran-Zan Wang, «Image hiding by optimal LSB subdtitutionand genetic algorithm,» *Pattern recognition,* p. 13, 1999.

[14] I. S. N. Ing. Gustavo Rodríguez, «Esteganografía: sustitución LSB 1 bit utilizando Matlab,» *Universidad Nacional de San Juan,* p. 6.

[15] J. J. E. O. Enrique Sánchez Acosta, «Aleatoriedad,» *Área de innovación y desarrollo, S.L.,* vol. 3, nº 1, p. 16, 2014.

[16] UAM, «arantxa.ii.uam.es,» [En línea]. Available: http://arantxa.ii.uam.es/~asuarez/docencia/master/webTS_2011/T01_numerosAleatorios_2x.pdf. [Último acceso: 13 Julio 2017].

[17] J. M. V. R. Adolfo Tamayo Briceño, «Generación de números aleatorios,» p. 9, 2014.

Anexo A Transmisor Tx con aleatorización

```
clear all;
clc;
c = imread('lenna_gris.png');
message                                                                    =
'ABCDEFGHIJKLMNOPQRZ12345678910ABCDEFGHIJKLMNOPQRZ12345678910!"#$%&/()()()()()ABCDE
FGHIJKLMNOPQRZ1234';
 %   message = '- -Esta es una prueba de esteganografia a traves de substitucion de
bit menos significativo o LSB- -';

    message = strtrim(message);
    m = length(message) * 8;
    AsciiCode = uint8(message);
    binaryString = transpose(dec2bin(AsciiCode,8));
    binaryString = binaryString(:);
    N = length(binaryString);
    b = zeros(N,1); %b is a vector of bits
    for k = 1:N
        if(binaryString(k) == '1')
            b(k) = 1;
        else
            b(k) = 0;
        end
    end
    s = c;
    height = size(c,1);
    width = size(c,2);
    k = 1;
    rng(1,'twister');                  % aleatorizacion de la posicion
    pos_h = randi(height,1,N+1);
    pos_w = randi(width,1,N+1);
    for l = 1:N
            i = pos_h(l);
            j = pos_w(l);
             LSB = mod(double(c(i,j)), 2);
            if (k>m || LSB == b(k))
                s(i,j) = c(i,j);
            else
                if (LSB==1)
                    s(i,j) = c(i,j) - 1;
                else
                    s(i,j) = c(i,j) + 1;
                end
            end
            k = k + 1;
    end
     figure(1)
     imshow(c)
     figure(2)
     imshow(s)
     imwrite(s, '111.png');
```

```
    %% SNR computation
    Power_c_image     = 0;
    Power_error_image = 0;

       for i = 1 : height
         for j = 1 : width
             dum_c     = double (c(i,j));
             dum_s     = double (s(i,j));
             dum_noisy = dum_c-dum_s;
             Power_c_image     = Power_c_image + dum_c.^2;
             Power_error_image = Power_error_image + (dum_noisy.^2);
         end
       end

       SNR_db = 10 * log10(Power_c_image/Power_error_image)
```

Anexo B Receptor Rx con aleatorización

```
    s = imread('111.png');
    height = size(s,1);
    width = size(s,2);
    %For this example the max size is 100 bytes, or 800 bits, (bytes * = bits
    m = N;
    k = 1;
    rng(1,'twister');
    pos_h = randi(height,1,N+1);
    pos_w = randi(width,1,N+1);
   for l = 1:N
            i = pos_h(l);
            j = pos_w(l);
            if (k <= m)
                br(k) = mod(double(s(i,j)),2);
                k = k + 1;
            end
    end
    binaryVector = br;
     binValues = [ 128 64 32 16 8 4 2 1 ];
     binaryVector = binaryVector(:);
     if mod(length(binaryVector),8) ~= 0
         error('Length of binary vector must be a multiple of 8.');
     end
     binMatrix = reshape(binaryVector,8,100);
     textString = char(binValues*binMatrix);
     disp(textString);
bt = b';
Errores = sum (br-bt)
```

Anexo C Transmisor Tx sin aleatorización

```
clear all;
clc;
c = imread('lenna_gris.png');
%   message =
'ABCDEFGHIJKLMNOPQRZ12345678910ABCDEFGHIJKLMNOPQRZ12345678910!"#$%&/()()()()()ABCDEFGHIJKLMNOPQRZ1234';
    message = '- -Esta es una prueba de esteganografia a traves de substitucion de bit menos
significativo o LSB- -';
    message = strtrim(message);
    m = length(message) * 8;
    AsciiCode = uint8(message);
    binaryString = transpose(dec2bin(AsciiCode,8));
    binaryString = binaryString(:);
    N = length(binaryString);
    b = zeros(N,1); %b is a vector of bits
    for k = 1:N
        if(binaryString(k) == '1')
            b(k) = 1;
        else
            b(k) = 0;
        end
    end
    s = c;
    height = size(c,1);
    width = size(c,2);
    k = 1;
    for i = 1 : height
        for j = 1 : width
             LSB = mod(double(c(i,j)), 2);
            if (k>m || LSB == b(k))
                s(i,j) = c(i,j);
            else
                if (LSB==1)
                    s(i,j) = c(i,j) - 1;
                else
                    s(i,j) = c(i,j) + 1;
                end
            end
            k = k + 1;
        end
    end
     figure(1)
     imshow(c)
     figure(2)
     imshow(s)
     imwrite(s, '111.png');
```

Anexo D receptor Rx sin aleatorización

```
    s = imread('111.png');
    height = size(s,1);
    width = size(s,2);
    %For this example the max size is 100 bytes, or 800 bits, (bytes * = bits
    % m = N;
    m = 800;
    k = 1;
    for i = 1 : height
        for j = 1 : width
            if (k <= m)
                br(k) = mod(double(s(i,j)),2);
                k = k + 1;
            end
        end
    end
    binaryVector = br;
     binValues = [ 128 64 32 16 8 4 2 1 ];
```

```
    binaryVector = binaryVector(:);
    if mod(length(binaryVector),8) ~= 0
        error('Length of binary vector must be a multiple of 8.');
    end
    binMatrix = reshape(binaryVector,8,100);
    %display(binMatrix);
    textString = char(binValues*binMatrix);
    disp(textString);
bt = b';
Errores = sum (br-bt)
```