Instituto Politécnico Nacional

Escuela Superior de Ingeniería Mecanica y Electrica

Unidad Culhuacan

Ingeniería en Computación

REPORTE TÉCNICO

Desarrollo de una API REST para la Protección de Imágenes de Color mediante Técnicas de Esteganografía y Cifrado

---

Presenta

Rodrigo Emmanuel Flores Avalos flores.rodrigo.emmanuel.dev@gmail.com

Directores del proyecto:

Dr. Manuel Cedillo Hernández

24 de mayo de 2023

Índice

1. Resumen

Este proyecto tiene como objetivo desarrollar una API REST pública que brinda protección a imágenes de color mediante técnicas de esteganografía y cifrado. La API permitirá a los usuarios incrustar información de color dentro de las imágenes utilizando técnicas de sustitución e interpolación. Luego, la imagen resultante se cifra mediante el algoritmo AES y mediante el uso de la criptografía de clave pública permite que las imágenes se compartan de manera segura. En resumen, la API proporciona una solución para proteger las imágenes de color mediante la combinación de técnicas de esteganografía y cifrado, lo que garantiza la confidencialidad y la integridad de las imágenes compartidas.

Palabras Clave: Criptografía, Estándar de Encriptación Avanzada, Esteganografía, Imágenes digitales de color Seguridad de la información.

2. Introducción

La seguridad de la información se ha convertido en un aspecto crucial en la actualidad, y la protección de los datos es una preocupación constante para muchas personas y organizaciones. En particular, las imágenes digitales de color contienen información valiosa y confidencial que debe ser protegida de posibles amenazas tanto en interceptación, como en manipulación de terceros en la nube. En este contexto, la criptografía [1] y la esteganografía [2] se han convertido en técnicas fundamentales para la protección de datos. La criptografía es el proceso de transformar la información en un formato ilegible sin la clave de descifrado correspondiente, mientras que la esteganografía permite ocultar información dentro de un archivo multimedia, como una imagen. En este proyecto de investigación, se propone el desarrollo de una API REST[3] que utiliza técnicas de esteganografía y cifrado para proteger imágenes de color. Esta API se basará en el Estándar de Encriptación Avanzada[4] y en técnicas de criptografía de clave pública para garantizar la confidencialidad y la integridad de las imágenes compartidas. En resumen, este trabajo de investigación aborda la importancia de la seguridad de la información en el contexto de las imágenes digitales de color, y propone una solución para su protección mediante la combinación de técnicas de criptografía y esteganografía.

3. Planteamiento del Problema

La transmisión de imágenes por medio de la web, ya sea persona a persona o a través de la nube, puede plantear serios problemas de privacidad y seguridad ya que no se tiene certeza de que personas ni con que intenciones tienen acceso a la información. Aunque existen contratos con los clientes como los términos y condiciones no se tiene certeza de quien pueda extraer o manipular esa información, debido a que al ser un servicio gratuito no se tiene el control total de los datos.Además, los métodos de cifrado tradicionales no son suficientes para proteger las imágenes digitales de color de posibles ataques y manipulaciones, ya que los algoritmos de cifrado no siempre son efectivos contra los ataques de fuerza bruta y pueden ser vulnerables a los ataques de hacking sofisticados. Algo importante para resaltar es que la mayoría de los métodos de cifrado se enfocan en proteger la integridad y la confidencialidad de los datos, pero no en ocultar la existencia de la información en sí misma. En otras palabras, aunque la imagen pueda estar cifrada, la presencia de información sensible dentro de la imagen puede ser detectada fácilmente mediante técnicas de análisis de esteganografía. Por lo tanto, es necesario contar con medidas de seguridad adicionales, como la criptografía, para proteger la información sensible contenida en las imágenes digitales de color durante la transmisión y el almacenamiento en línea. La esteganografía permite ocultar información dentro de las propias imágenes de color, lo que dificulta su detección y proporciona una capa adicional de protección contra posibles amenazas.

4. Objetivos

4.1. Objetivo General

El objetivo general de este proyecto de investigación es ofrecer una solución para proteger imágenes de color mediante técnicas de esteganografía y cifrado. Para ello, se desarrolló una API REST pública, utilizando lenguaje Python[5] y su framework Fast API[6], que permita a los usuarios proteger imágenes de color incrustando

información oculta en ellas mediante técnicas de esteganografía y, posteriormente, cifrándolas con el algoritmo AES para asegurar su seguridad.

Además, se busca que esta API sea de acceso público e implementable para cualquier desarrollador interesado en aplicaciones enfocadas en la seguridad de la información, permitiendo así que se implementen soluciones más seguras y confiables en el manejo de imágenes de color. Con esta solución, se espera brindar mayor protección a las imágenes digitales, evitando que terceros no autorizados puedan acceder a información confidencial contenida en ellas y enfocando el desarrollo en la privacidad y seguridad de los datos.

4.2. Objetivos específicos

1. Definir una estructura de comunicación asíncrona entre las diferentes entidades del cifrado y descifrado.
2. Elaborar un algoritmo web Backend[7] para la recepción de imágenes con Fast API.
3. Diseñar un algoritmo en Python que oculte la información de color dentro de la imagen.
4. Diseñar un algoritmo en Python que tome la imagen con la información de color ocultada y sea cifrada mediante el algoritmo AES.
5. Diseñar un algoritmo que pueda descifrar la imagen y extraer la información de color para unirla con la componente Y obtenida por el descifrado.

5. Límites y alcances

5.1. Límites

1. Para enviar y recibir los datos de cifrado y descifrado se requerirá de una conexión a internet de mínimo 4mb/s (3G)*.
2. El sistema efectuará el cifrado si y solo si la clave pública es correctamente proporcionada.
3. Para el descifrado de información se requerirá una clave privada proporcionada al momento del cifrado de la información.
4. La API solo será capaz de cifrar imágenes de color con formatos PNG, JPG y BMP.

5.2. Alcances

1. Para que la imagen sea devuelta con la información de color debe haber sido cifrada por la API.
2. La API debe tener la capacidad de interactuar con cualquier sistema cliente que envíe peticiones POST con imágenes en formato JPG, PNG y BMP.
3. La API validará las peticiones para garantizar la estructura de datos recibida y poder iniciar los procesos de cifrado y descifrado.
4. La API podría no recibir la información de descifrado en un JSON debido a que esta información se contiene en la imagen.

6. Justificación

La privacidad y la seguridad de la información son preocupaciones clave y constantes en el mundo digital actual. La transmisión de imágenes por medio de la web, ya sea entre individuos o a través de la nube, es especialmente vulnerable a la manipulación y el acceso no autorizado por terceros malintencionados.

La utilización de técnicas de esteganografía y cifrado es una manera efectiva de ocultar información confidencial y protegerla de posibles ataques ya que se requiere saber dónde está ocultada la información de color dentro de la imagen descifrada aún en el hipotético caso de que el cifrado AES sea vulnerado o la clave de descifrado sea expuesta por algún atacante. En particular, la estenografía permite ocultar información dentro

de imágenes de color sin modificar el aspecto visual de la imagen, lo que la hace ideal para proteger información confidencial contenida en imágenes de color. Además, el uso del algoritmo AES en el cifrado de la información garantiza un alto nivel de seguridad en la transmisión de datos.

El desarrollo de una API REST con estas características permitiría a los desarrolladores ofrecer una mayor protección y seguridad de la información a los usuarios de aplicaciones que involucren el manejo de imágenes de color, lo que puede resultar de gran valor en ámbitos como la medicina, la seguridad nacional, la justicia, la banca y el comercio electrónico.

7. Estado del Arte

Autor(es)	**Tema**	**País**
Osama Hosam, Muhammad Ahmed. *Taibah University.* (Publicación 2019)	Hybrid Design for Cloud Data Security Using Combination of AES, ECC and LSB-Steganography[8].	Arabia Saudita
Rose Adee. *Stockholm University.* (Paper 2022)	A Dynamic Four-Step Data Security Model for Data in Cloud Computing Based on Cryptography and Steganography[9].	Suiza
Haider TH. Salim ALRikabi, Hussein Tuama Hazim *Wasit University and University of Misan* (Paper 2022)	Enhanced Data Security of Communication System Using Combined Encryption and Steganography [10].	Iraq

Tabla 1: Tabla que refleja una investigación del estado tecnológico en el mundo sobre investigaciones que abordan el mismo caso de estudio

En el proyecto de Osama Hosam y Muhammad Ahmed [8]. presenta un método de seguridad híbrido para proteger los datos en la nube y un modelo que se adapta a los requisitos de seguridad y eficiencia del diseño.

En el proyecto de Rose Adee [9] se describe las tecnologías, así como la interacción entre las técnicas y algoritmos de cifrado y esteganografía para poder asegurar datos en la nube.

En el proyecto de Haider TH. Salim ALRikabi y Hussein Tuama Hazim [10] se describe un enfoque para mejorar la seguridad de imágenes usando técnicas de esteganografía y criptografía. El objetivo del estudio es proporcionar una solución más segura y eficiente para proteger los datos confidenciales.

8. Marco Teórico

8.1. Modelo de Madurez de Leonard Richardson para API REST

El modelo de madurez de Richardson, también conocido como nivel de madurez de Richardson, es una guía útil para diseñar y desarrollar API RESTful. Este modelo se basa en la idea de que una API RESTful puede tener diferentes niveles de madurez en cuanto a su diseño y funcionalidad, y por lo tanto, establece niveles de madurez que una API puede alcanzar.

El modelo de madurez de Richardson consta de cuatro niveles, cada uno de los cuales representa un paso en el camino hacia una API RESTful más completa y madura. El nivel 0 representa una API que no cumple

con los principios fundamentales de REST, mientras que el nivel 3 representa una API RESTful completamente madura. Este modelo también enfatiza en la importancia de los recursos y su representación en la API, lo que resulta en una API bien estructurada y fácil de entender para los desarrolladores.

Se eligió el modelo de madurez de Richardson porque proporciona una guía clara y bien definida para el diseño y desarrollo de API RESTful, y permite que los desarrolladores y diseñadores sigan un camino claro hacia la creación de una API RESTful madura y completa. Además, el modelo de madurez de Richardson es ampliamente utilizado en la industria y ha sido adoptado por muchas empresas líderes en tecnología debido a su enfoque gradual y evolutivo para el diseño de API RESTful.

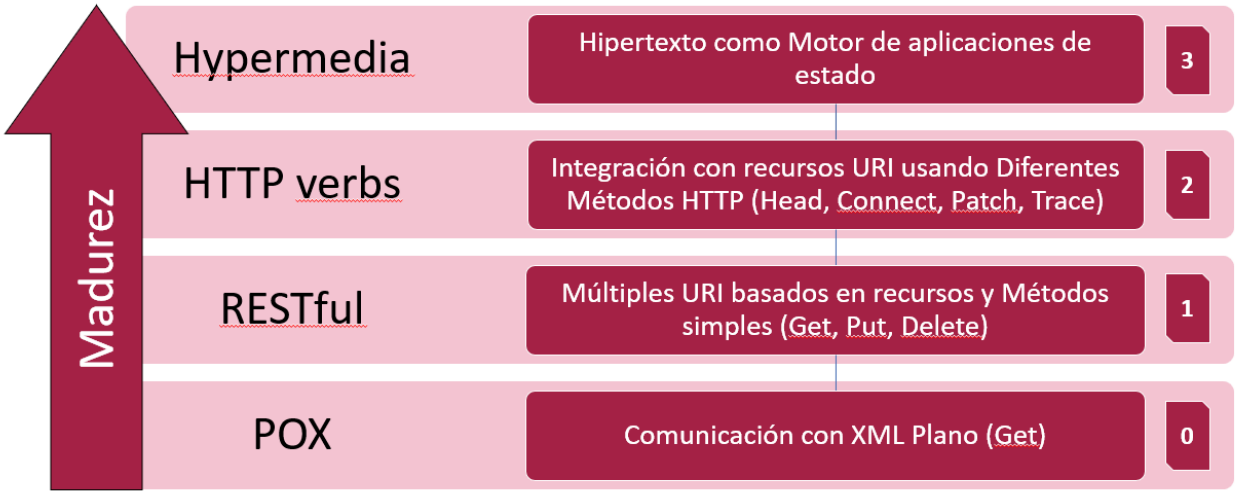

Figura 1: Diagrama del modelo de madurez por niveles de Leonard Richardson para API REST

8.2. FastAPI

La elección de FASTAPI como tecnología para el desarrollo de esta API REST se basó en una serie de factores. En primer lugar, FASTAPI es una de las tecnologías más rápidas y eficientes para la creación de APIs en Python, lo que la hace ideal para proyectos que requieren un alto rendimiento y velocidad. Además, su sintaxis es escalable y enfocada a los micro-servicios, lo que hace eficiente la construcción de aplicaciones que cumplan con un buen ciclo de DevOps.

Otro factor clave en la elección de FASTAPI es su integración con otros frameworks y herramientas comunes en el mundo de la programación en Python, como Pydantic y Starlette. Estas integraciones permiten la validación de datos y una mayor eficiencia en el manejo de solicitudes y respuestas HTTP.

8.3. Open CV

OpenCV es una biblioteca de procesamiento de imágenes y visión por computadora de código abierto que se utiliza ampliamente en la industria y la investigación para aplicaciones de procesamiento de imágenes y video. Se eligió OpenCV como tecnología en Python debido a su amplia variedad de algoritmos de procesamiento de imágenes, lo que permite la implementación de técnicas de esteganografía y cifrado en las imágenes de manera eficiente y precisa. Además, OpenCV es compatible con una amplia gama de lenguajes de programación, lo que lo hace muy versátil para aplicaciones que requieren el procesamiento de imágenes.

8.4. Esteganografía: LSB

El método de ocultamiento de datos LSB (Least Significant Bit) es una técnica común de esteganografía que consiste en ocultar datos en los bits menos significativos de una imagen sin alterar significativamente su apariencia visual.

En comparación con otras técnicas de esteganografía, también permite ocultar una cantidad relativamente grande de datos sin afectar la calidad visual de la imagen, lo que es importante para evitar que terceros detecten la presencia de información oculta en la imagen ya que esta técnica aprovecha el hecho de que los datos digitales se almacenan en una imagen en forma de bits, y que los bits menos significativos de los valores de píxel tienen menos impacto en la calidad de la imagen. Al ocultar los datos en los bits menos significativos de los valores de píxel, el cambio en la calidad de la imagen es mínimo y difícil de detectar visualmente lo que la hace atractiva para aplicaciones prácticas como la protección de información en imágenes. Aunque existen técnicas más sofisticadas que pueden ser más efectivas en la ocultación de datos, estas pueden ser más complejas y requerir más tiempo y recursos para implementar, lo que las hace menos adecuadas para aplicaciones prácticas.

8.5. Cifrado AES

El cifrado AES (Advanced Encryption Standard) es un algoritmo de cifrado simétrico ampliamente utilizado en la protección de datos, que utiliza bloques de datos de 128 bits y claves de cifrado de diferentes tamaños. Entre los diferentes modos de cifrado disponibles en AES, se eligió el modo OFB (Output Feedback) debido a su capacidad de cifrar y descifrar datos en bloques independientes sin la necesidad de una reconstrucción completa de la cadena de cifrado.

El modo OFB utiliza una matriz de retroalimentación de salida para convertir una clave secreta y un vector de inicialización en una secuencia de clave pseudo-aleatoria, que se utiliza para cifrar los datos en bloques independientes. Este modo de cifrado tiene la ventaja de ser paralelizable y resistente a errores en la transmisión de datos.

Además, el modo OFB de AES proporciona una mayor seguridad en la transmisión de datos, ya que cada bloque cifrado es independiente de los demás bloques, lo que dificulta la tarea de cualquier tercero no autorizado que intente acceder a la información. Además, este modo de cifrado proporciona una alta velocidad de cifrado y descifrado, lo que es esencial para el procesamiento de imágenes grandes.

El modo OFB, en particular, proporciona una mayor seguridad al cifrar imágenes ya que evita el problema de la propagación de errores en cascada que puede ocurrir con otros modos de cifrado, como el modo CBC. Esto significa que cualquier error que ocurra durante la transmisión o el cifrado de una sección de la imagen no afectará las secciones posteriores.

En general, el uso del cifrado AES en modo OFB proporciona una solución segura y eficiente para proteger imágenes de color mediante técnicas de cifrado, lo que lo convierte en una elección ideal para el proyecto.

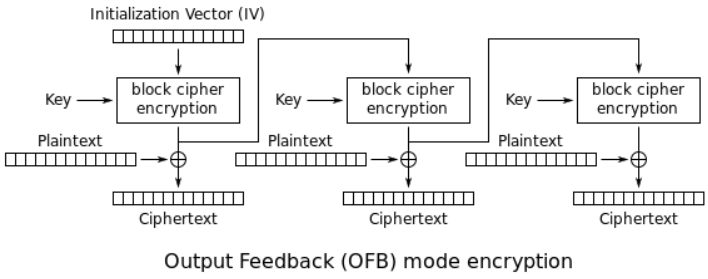

Figura 2: Diagrama del proceso de cifrado del modo Output Feedback (OFB)

9. Desarrollo

9.1. Diagrama de funcionamiento

La API va a exponer diversos puertos a través de los Endpoints públicos donde se vaya a desplegar el contenedor de la aplicación, para poderse comunicar se va a hacer una respuesta por petición lo que implicaría que por cada solicitud post con una imagen válida se responderá con su parte correspondiente,

La recepción de solicitudes validará si la imagen será descifrada o cifrada, si la imagen se manda descifrar primero se verificará que esta imagen esté cifrada, si al descifrar la imagen se puede encontrar la información de descifrado oculta dentro del resultante de la imagen descifrada esta podrá ser restaurada, en caso contrario

no se podrá descifrar porque implicaría que la imagen proporcionada no fue cifrada por la API aun cuando esta esté en el espacio de la escala de grises o canal de luminancia.

Si la imagen se manda cifrar se validará que primeramente que esté dentro de los formatos soportados (JPG, PNG y BMP), en segunda instancia se validará que no sea una imagen que posea canales diferentes a tres ya que si se posee un cuarto canal por ejemplo el canal de transparencia en el caso del formato PNG este no será considerado para la API, si la imagen posee menos de tres canales significa que la imagen será binaria o está en un espacio de color distinto por lo que tampoco estará considerada.

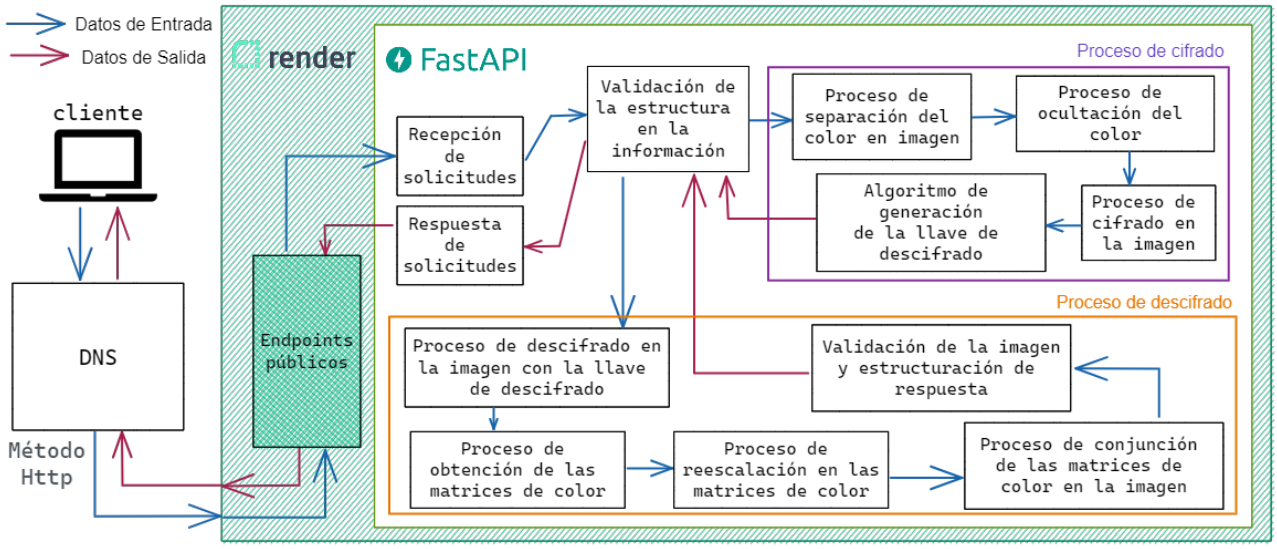

Figura 3: Diagrama del flujo de la información interna de la API

9.2. Encriptar y ocultar

La primer etapa es el ocultamiento de la información de color y la segunda la encriptación. En esta primera etapa se transforma a imagen del espacio de color RGB hacia el espacio de color YCbCr lo que proporciona de capas de color interdependientes, se toma la imagen del canal de luminancia Y y se le incrusta la información del resultado de la interpolación de la información de las componentes de Crominancia Azúl o Cb y de la Crominancia Roja o Cr, por lo que solo se ocupa la mitad de la imagen para ocultar esta información interpolada, a esto se le agrega un archivo en formato JSON con la información de dónde está ocultada la información de Cb y Cr y cuales eran las dimensiones originales de la imagen para poder posteriormente reconstruirla.

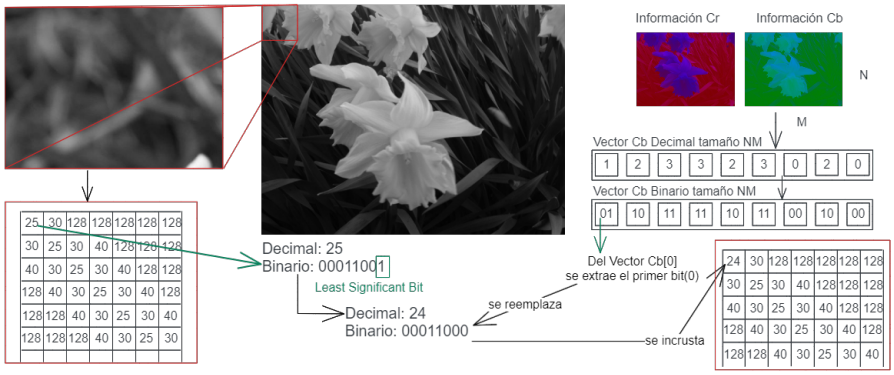

Figura 4: Diagrama del flujo de la información para el ocultamiento de la información

En la figura 5 se puede apreciar claramente el flujo de trabajo de este proceso y la consolidación en la información del JSON que será ocultado en la imagen del canal Y, el dato de entrada es una imagen RGB, posteriormente se observa el conjunto Ïmagen YCbCr"que representa la imagen convertida a los canales de color Luminancia (Y), Crominancia Azul (Cb) y Crominancia Roja (Cr). Continuadamente se muestra el escalamiento de las imágenes Cb y Cr y se representa gráficamente el ocultamiento en la imagen Y.

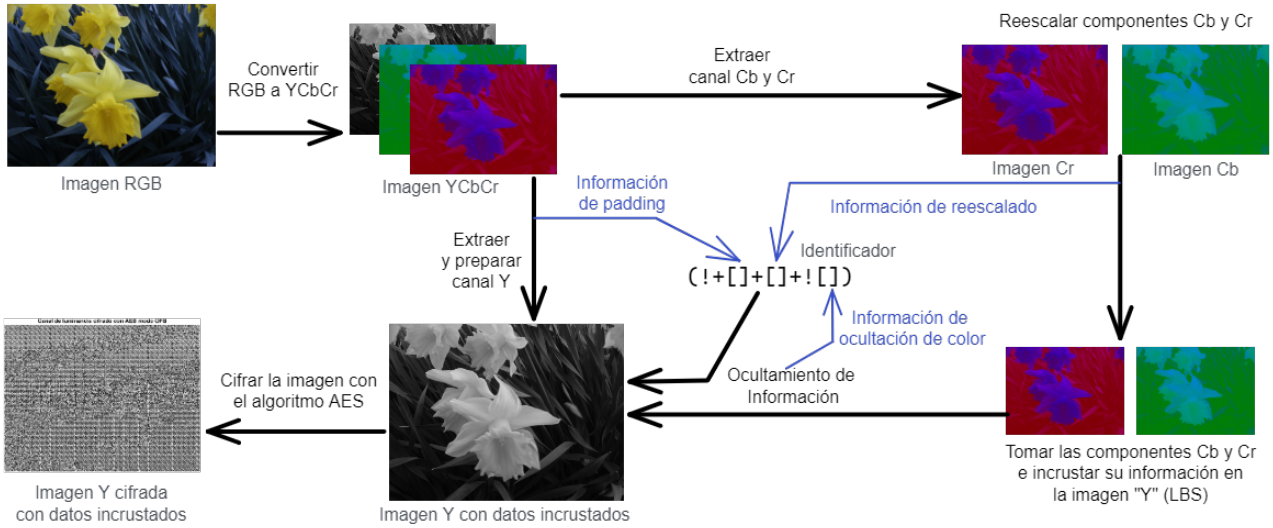

Figura 5: Diagrama del flujo de la información para el cifrado

Lo ultimo que queda pendiente es el cifrado por el algoritmo AES en su modo OFB que es un proceso que requiere que la matriz de la imagen sea múltiplo de 16 por lo que se agrega un padding con información en cero para poder identificarlo después de la desencriptación.

9.3. Desencriptar y desocultar

Para poder restaurar la imagen primero se tiene que desencriptar, después de esto la API buscará la cadena que represente la huella de la api que se incrusta al final del JSON, si esta cadena se encuentra se podrá continuar con el proceso de lectura de la información de color ocultada (Esto hace referencia a la clave privada del sistema de clave pública que requiere de una llave pública y una llave privada).

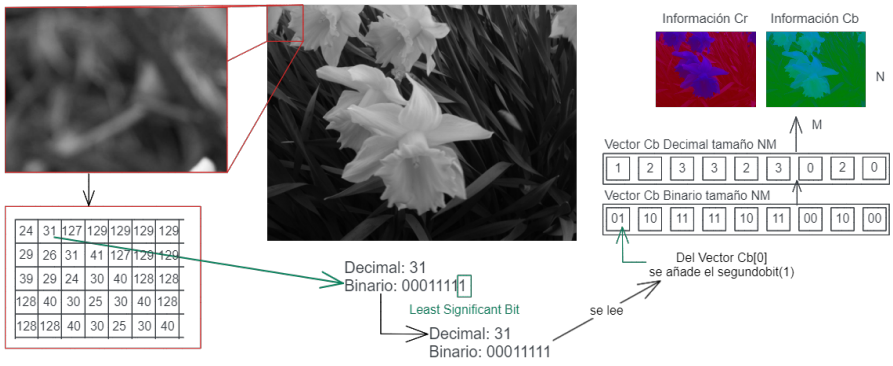

Figura 6: Diagrama del flujo de la información para el cifrado

Después de tener la matriz de datos de color tanto Cb y Cr se extrapolan para poder tener matrices de color del mismo tamaño de la imagen Y desencriptada que corresponde al canal de luminancia dentro del espacio YCbCr. Después se junta esa información para completar una imagen con tres canales de color en el espacio YCbCr para poderla transformar a RGB y terminar con una imagen restaurada.

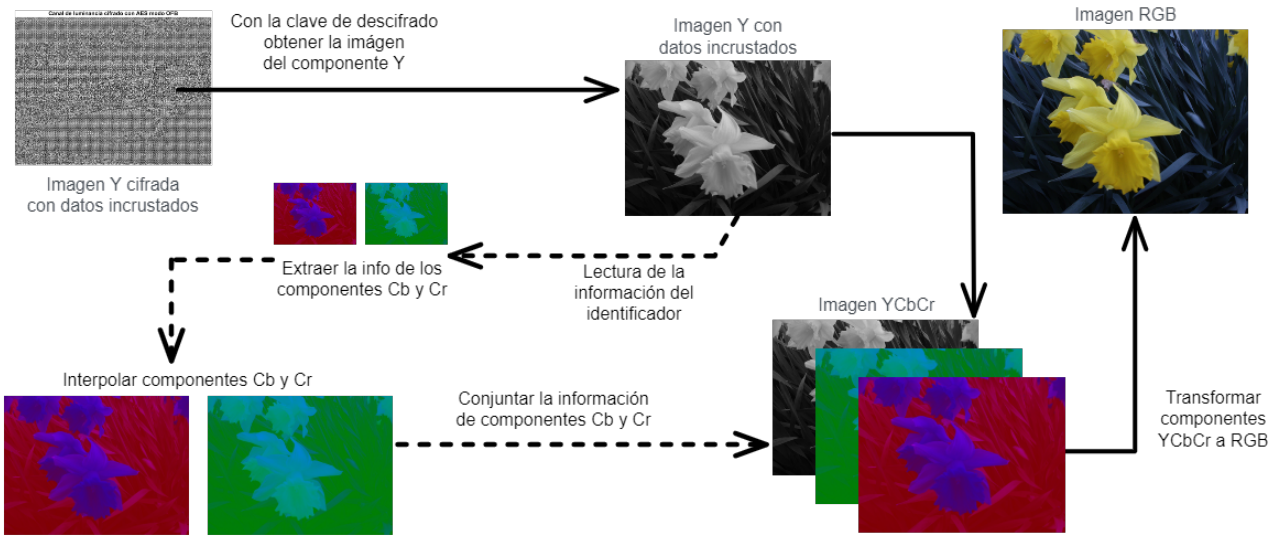

Figura 7: Diagrama del flujo de la información para el cifrado

10. Resultados

10.0.1. División en componentes

La API debe recibir una solicitud POST en su endpoint /API/Encryptçon un header "multipart/form-dataz una imagen que cumpla las siguientes condiciones:

1. La imagen posea tres canales de color sin transparencias
2. La imagen está en uno de las extenciones válidas (PNG, JPG o BMP)

Por ejemplo la API recibe la siguiente imagen correctamente:

Figura 8: Imagen RGB recibida por la solicitud POST

Para poder ocultar los componentes Cb y Cr dentro de la información del componente de luminancia Y se convierte el espacio de color RGB a YCbCr. Esta primera etapa de división de componentes se puede apreciar de la siguiente forma:

Figura 9: Imágenes resultantes de la división de componentes, se aprecia el canal Cr, el Cb y el canal Y

Para poder visualizar la diferencia entre Cr y Cb como lo vería el ojo humano de forma natural se puede combinar la imagen Cr con la componente Y y para completar los tres canales se llena el tercer canal con una matriz de ceros, este proceso se puede hacer con la componente Cb de la misma forma que con Cr.

Figura 10: Imágenes resultantes de la combinación de componentes, se aprecia el espacio Y0Cr y el espacio YCb0

10.0.2. Ocultamiento de la información

Para poder hacer el ocultamiento de las componentes Cb y Cr primero se necesita reducir la información con métodos algorítmicos como la interpolación bicúbica tanto para reducir los componentes de la imagen como para ampliar los componentes de la imagen. Del mismo modo se aplica un filtro de Antialising para poder tener una imagen que al momento de interpolarla a sus dimensiones originales pueda obtener una gama de píxeles más amplia debido a su desvanecimiento, este Antialising lo ocasiona un desenfoque con un filtro Gaussiano. En las siguientes imágenes se aprecia el canal Cb y el canal Cb pero con una interpolación bicúbica aplicada a las dimensiones en $n/4$ de largo y $m/4$ de ancho con un filtro Gaussiano aplicado

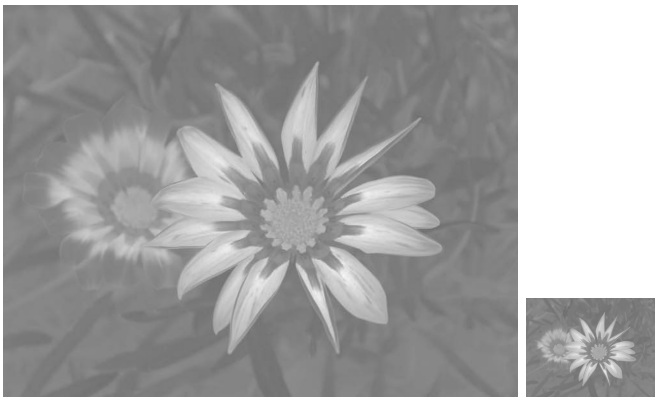

Figura 11: Imágenes resultantes de la reducción de componentes.

En el proceso de incrustación de información en el bit menos significativo de una imagen, no se produce un cambio visual significativo en la imagen original a simple vista. El bit menos significativo es el bit más débil en términos de la representación numérica de la información en la imagen, y por lo tanto, su cambio no afecta en gran medida la apariencia visual de la imagen.

Para evaluar la calidad de la imagen después de la incrustación de datos, se utiliza una métrica como el PSNR (Porción máxima de Señal a Ruido). El PSNR mide la relación entre la señal original y el ruido que se produce durante la incrustación de datos. Se utiliza para comparar la calidad de la imagen de la componente Y contrastada con la imagen de la componente Y con los canales Cb y Cr ocultos dentro de esa imagen.

En términos simples, el PSNR mide la cantidad de información que se ha perdido o añadido durante el proceso de incrustación de datos en la imagen. Cuanto mayor sea el valor del PSNR, mejor será la calidad de la imagen resultante después de la incrustación de datos.

Figura 12: Imágenes representantes del pre-ocultamiento y el post-ocultamiento de la información. PSNR: 51.9

Si comparamos el histograma de la imagen original con el histograma de la imagen resultante después de la incrustación de datos, podemos ver claramente si se han producido cambios en la información de la imagen.

En el caso de la incrustación de datos en el bit menos significativo de una imagen, es probable que no se produzcan cambios significativos en el histograma de la imagen. Sin embargo, si observamos de cerca ambos histogramas, podremos detectar pequeñas diferencias en las intensidades de los píxeles entre la imagen original y la imagen con los datos ocultos.

El hecho de que se produzcan cambios en los histogramas de ambas imágenes después del proceso de incrustación de datos indica que el proceso ha sido exitoso y que los datos han sido correctamente incrustados en la imagen.

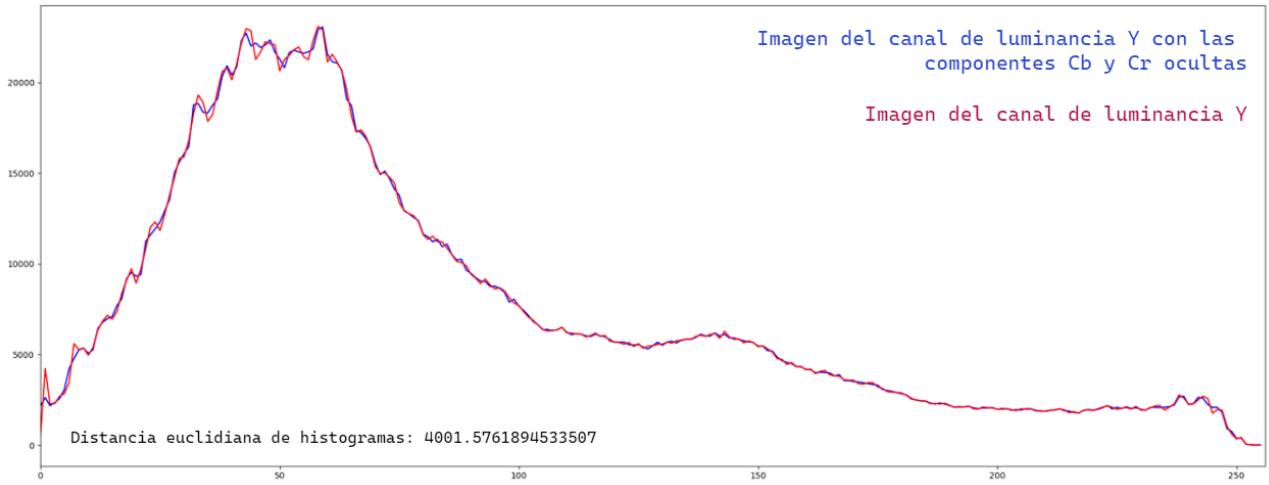

Figura 13: Comparación de histogramas de las imágenes, se aprecia en el histograma rojo la imagen original del canal de luminancia (Y) y en color azul el histograma del canal de luminancia (Y) pero con los datos de las componentes Cb y Cr ocultos dentro de esta imagen

10.0.3. Cifrado y descifrado de la imagen

Durante esta etapa del proceso, no se obtiene un resultado visualmente apreciable, ya que el resultado del cifrado es un criptograma que no es visible. Sin embargo, se realizan validaciones importantes durante el proceso de cifrado. Por ejemplo, se debe agregar información adicional a la imagen para asegurar que cumpla con el requisito del cifrado AES en modo OFB, que establece que los datos a cifrar deben ser múltiplos de 16 bytes o lo que es equivalente 128 bits, que es el tamaño del bloque utilizado para el cifrado. Asimismo, antes de aplicar el cifrado, es necesario preparar la imagen convirtiéndola en datos binarios para su posterior procesamiento. La imagen se divide en bloques de datos del tamaño requerido por el algoritmo AES y se realiza un relleno para asegurar que los bloques tengan el tamaño adecuado.

Posteriormente, se genera una clave maestra de cifrado. A partir de esta clave maestra, se realiza una expansión de claves para generar las subclaves necesarias para el modo OFB.

El proceso de cifrado comienza con la inicialización de un vector de realimentación. Luego, se aplica el algoritmo AES a cada bloque de datos de la imagen utilizando la subclave correspondiente, generada durante la expansión de claves. El resultado de cada bloque cifrado se obtiene mediante la operación XOR entre el bloque de datos original y la salida del algoritmo AES. El vector de realimentación se actualiza utilizando la misma subclave y se repite el proceso para cada bloque de datos.

El descifrado de la imagen cifrada se realiza de manera similar al cifrado. Se inicializa un vector de realimentación y se aplica el algoritmo AES inverso a cada bloque cifrado utilizando la subclave correspondiente. Nuevamente, se utiliza la operación XOR para obtener el bloque de datos descifrado. El vector de realimentación se actualiza y se repite el proceso para cada bloque cifrado.

10.0.4. Desocultamiento de la información de color

Durante el proceso de des-ocultamiento, se lleva a cabo la extracción de los canales de crominancia que fueron previamente ocultados en la imagen del canal de luminancia Y, el cual se obtiene mediante el descifrado del criptograma. Una vez obtenidas estas matrices, se procede a realizar la interpolación de dichas matrices para obtener matrices del mismo tamaño que la imagen del canal de luminancia. Durante este proceso también se le quita el Padding a la imagen del canal Y que se usó para poder cifrar en el algoritmo AES en su modo OFB.

En la figura 14 se muestra un resultado gráfico de este proceso.

Por ultimo ya que las matrices fueron interpoladas al alto y ancho correspondientes a la imagen Y, se procede a unir los canales de crominancia para poder tener como resultado una imagen en el espacio de color YCbCr, a esta imágen solo faltaría poder realizar un cambio en el espacio de color de YCbCr a RGB, el cual se puede verificar en la Figura 15.

Figura 14: Imágenes resultantes de la interpolación de matrices.

Figura 15: Imágenes resultantes de la conjunción de crominancias y de la transformación del espacio de color. PSNR: 47.43 dB

11. Conclusiones

El desarrollo de este proyecto representa un hito inicial en una serie de avances necesarios para garantizar la seguridad de las imágenes en color tanto en la nube como en su transmisión. Aunque se han promulgado leyes como la Ley Olimpia en México, lamentablemente aún no existen soluciones tecnológicas que aborden de manera integral la raíz del problema. Este proyecto tiene como objetivo proporcionar una base para que las futuras generaciones puedan desarrollar soluciones en el ámbito de la seguridad de la información y comprender verdaderamente el impacto que esto puede tener en aplicaciones como WhatsApp. Aunque esta aplicación cifra la comunicación de extremo a extremo, la transmisión de imágenes dentro de la misma aplicación puede ser más permisiva. Con este proyecto, se pretende generar un impacto en el campo del desarrollo y la investigación, inspirando la creación de más soluciones que incluyan la transmisión de imágenes cifradas.

Referencias

[1] P. J. Paar C., *Understanding cryptography: a textbook for students and practitioners.* Springer Science and Business Media, 2009.

[2] X. W. M. B. Shi Y. Q., Li X. Zhang, *Reversible data hiding: advances in the past two decades.* IEEE access, 4, 3210-3237., 2016.

[3] L. Li and W. Chou, *Design and Describe REST API without Violating REST: A Petri Net Based Approach,.* IEEE International Conference on Web Services, Washington, DC, USA, 2011.

[4] B. Schneier, *Applied cryptography: protocols, algorithms, and source code in C.* john wiley and sons, 2007.

[5] S. J. Nielson and C. K. Monson, *Practical Cryptography in Python: Learning Correct Cryptography by Example.*, 2019.

[6] M. Lathkar, *Introduction to FastAPI. In High-Performance Web Apps with FastAPI: The Asynchronous Web Framework Based on Modern Python.* Berkeley, CA: Apress., 2023.

[7] J. R. M. F. F. L. Pérez Ibarra, S. G. Quispe, *Herramientas y tecnologías para el desarrollo web desde el FrontEnd al BackEnd.* XXIII Workshop de Investigadores en Ciencias de la Computación (WICC 2021, Chilecito, La Rioja, 2021.

[8] . A. M. H. Hosam, O., *Hybrid design for cloud data security using combination of AES, ECC and LSB steganography.* International Journal of Computational Science and Engineering, 2019.

[9] R. Adee, *A Dynamic Four-Step Data Security Model for Data in Cloud Computing Based on Cryptography and Steganography.* Sensors, 2022.

[10] A. H. T. and H. H. T., *Enhanced data security of communication system using combined encryption and steganography.* International Journal of Interactive Mobile Technologies, 2021.