Instituto Tecnológico Superior de San Andrés Tuxtla.

Ingeniería Informática

Roberto Toto Librado

Luis Alberto Aguilar Rendon

Ángel de Jesús Fiscal Málaga

CIFRADOR HÍBRIDO RSA–AES CON REGISTRO EN BASE DE DATOS

Matrícula: IINF-2010-220

Profesor: Erick de Jesús Téllez.

San Andrés Tuxtla, Veracruz.

Noviembre 2025

DOCUMENTACIÓN COMPLETA DEL PROYECTO

CIFRADOR HÍBRIDO RSA–AES CON REGISTRO EN BASE DE DATOS

1. ESPECIFICACIÓN DE REQUERIMIENTOS

1.1 Definición General del Proyecto

El proyecto consiste en un sistema de cifrado híbrido que utiliza AES para cifrar archivos y RSA para cifrar la clave simétrica generada. El sistema registra en una base de datos las operaciones realizadas, incluyendo el nombre del archivo, la fecha y la clave cifrada.

1.2 Requerimientos Funcionales

- Generar claves RSA (pública y privada) de 2048 bits.
- Cargar claves RSA desde archivos PEM.
- Cifrar archivos utilizando AES en modo GCM.
- Cifrar la clave AES mediante RSA.
- Guardar la clave AES cifrada en un archivo .key.enc.
- Descifrar archivos utilizando la clave AES original.
- Registrar las operaciones en una base de datos MySQL.
- Interfaz gráfica mediante JavaFX.

1.3 Requerimientos No Funcionales

- Lenguaje: Java 17 o superior.
- Librerías: JavaFX, JDBC.
- Base de datos: MySQL local.
- Seguridad: uso de algoritmos modernos (RSA-OAEP SHA-256, AES-GCM).
- Portabilidad: compatible con Windows y Linux.

2. PROCEDIMIENTOS

2.1 Procedimiento de Desarrollo

El proyecto se desarrolló estructurando los componentes en paquetes:

- com.cifrador.utils: utilidades criptográficas.
- com.cifrador.encryptor: implementación del cifrado híbrido.
- com.cifrador.database: conexión y registro en BD.
- com.cifrador.ui: interfaz gráfica.

2.2 Procedimiento de Instalación

1. Instalar MySQL y crear la base de datos “cifrador”.
2. Crear tabla “operaciones”.
3. Ejecutar el proyecto Java con JavaFX configurado.
4. Cargar claves RSA o generar nuevas.
5. Seleccionar archivo y cifrar o descifrar.

2.3 Pruebas Realizadas

- Prueba de cifrado con archivos de texto.
- Prueba con archivos grandes.
- Prueba de integridad en descifrado.
- Verificación de registros en BD.

3. ARQUITECTURA DEL SISTEMA

3.1 Módulos Principales

- AESUtil: genera claves AES, IV y cifra/descifra contenido.
- RSAUtil: genera claves RSA, carga PEM y cifra/descifra claves AES.
- HybridCipher: coordina el proceso híbrido.
- DatabaseHelper: registra operaciones.
- MainApp: interfaz gráfica y la principal que manda a llamar las diferentes funciones del software

3.2 Flujo General del Sistema

1. Usuario crea su propio usuario para generar claves pública y privada
2. Se genera clave AES.
3. Se selecciona la clave pública
4. Se selecciona el archivo
5. Se cifra el archivo con AES-GCM.
6. Se cifra la clave AES con RSA.
7. Se guarda archivo cifrado y clave cifrada.
8. Se registra operación en BD.
9. Para descifrar: se recupera AES original usando RSA y se descifra archivo.

4. MODELO DE DATOS

Tabla: operaciones

- id_operacion (INT, PK)
- nombre_archivo (VARCHAR 255)
- fecha (DATETIME)
- clave_aes_cifrada (TEXT)
- usuario (VARCHAR 255)

5. PROCESOS PRINCIPALES

5.1 Cifrado

- AES genera clave de 256 bits.
- AES cifra archivo con IV de 12 bytes.
- RSA cifra clave AES.
- Se guarda archivo .enc y .key.enc.

5.2 Descifrado

- RSA descifra clave AES.
- AES descifra archivo original.

6. API DE MÉTODOS PRINCIPALES

HybridCipher.encryptFile()

Entrada: archivo original

Salida: archivo .enc y archivo .key.enc

Retorna: clave AES cifrada en Base64

HybridCipher.decryptFile()

Entrada: archivo cifrado .enc y .key.enc

Salida: archivo descifrado .dec

AESUtil.encryptFile()

AESUtil.decryptFile()

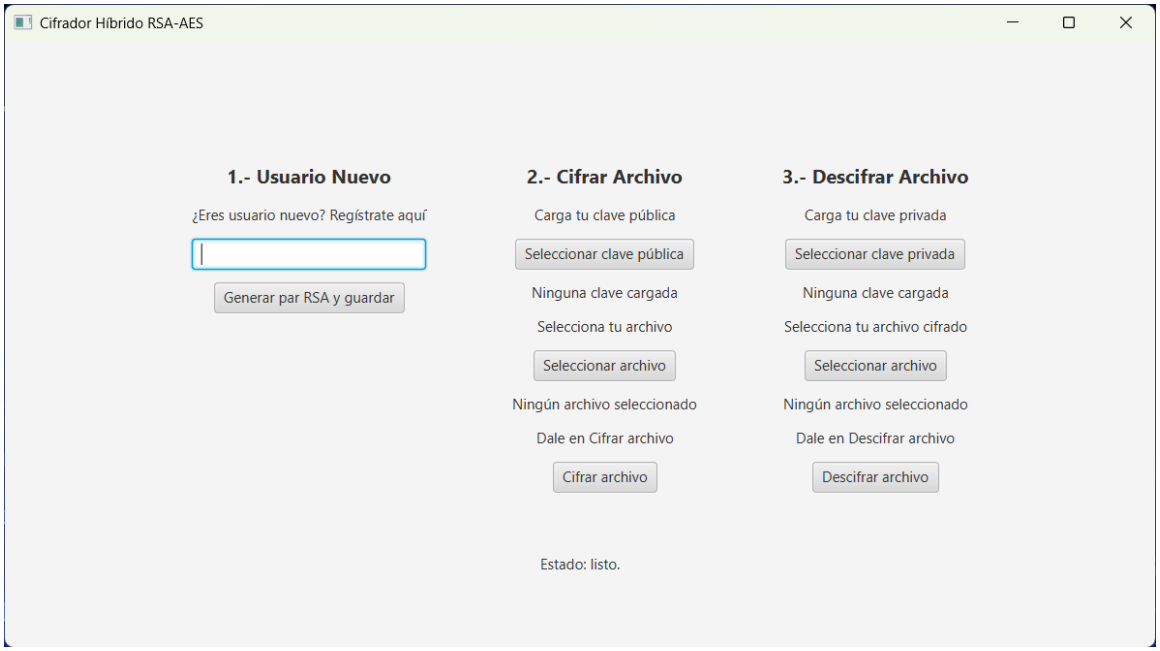
RSAUtil.generateKeyPair()

RSAUtil.loadPublicKey()

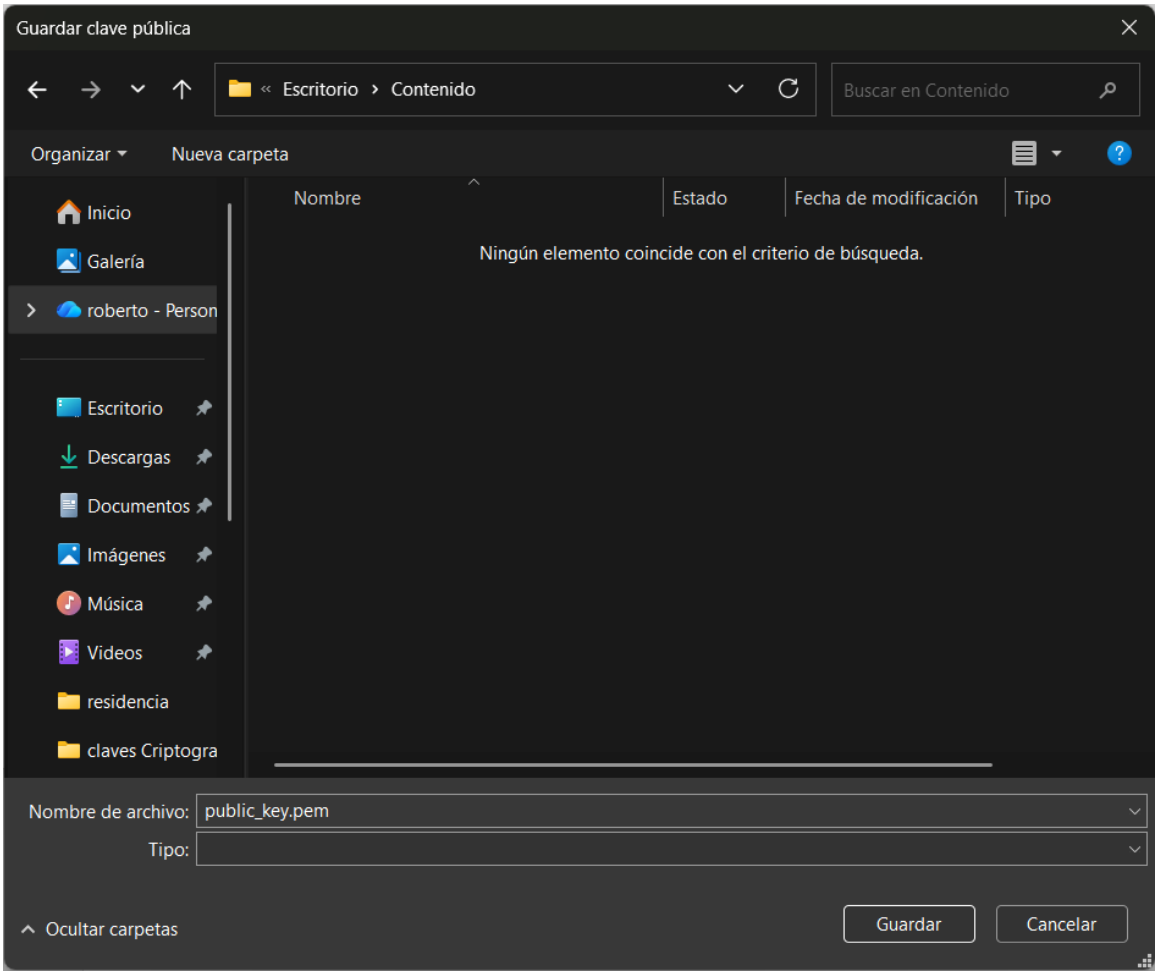
RSAUtil.loadPrivateKey()

6. COMO FUNCIONA

Accedemos al acceso directo de nuestro cifrador hibrido



Ahora crearemos un Usuario en este ejemplo será “Roberto” y seleccionamos generar par RSA y guardar

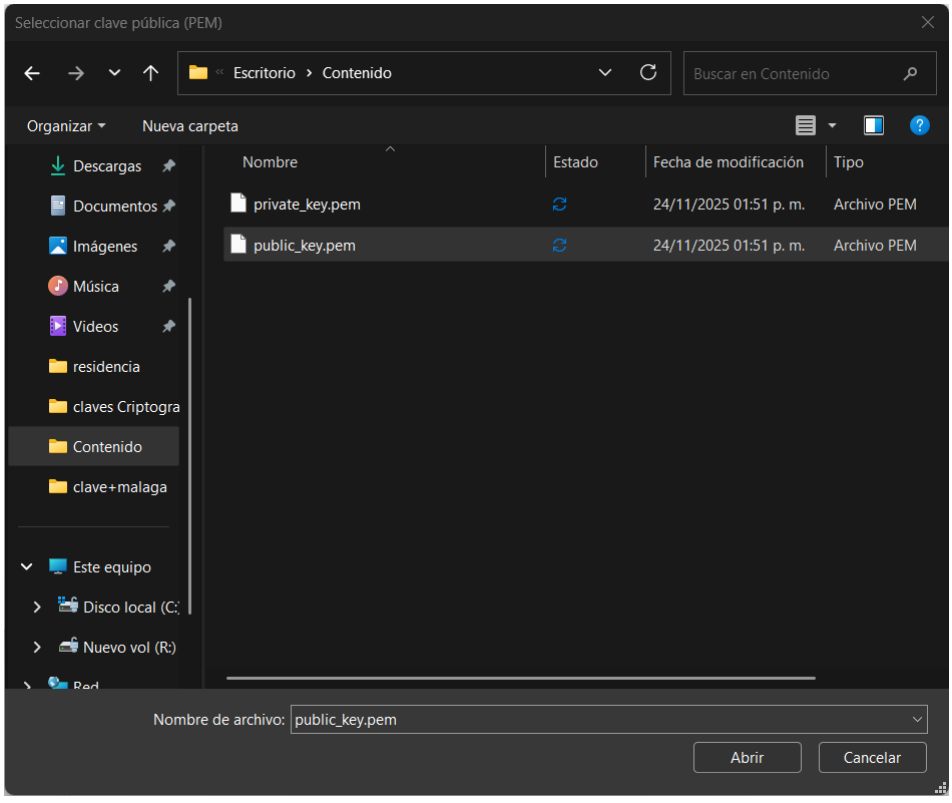


Nos generara dos llaves una publica y otra privada asegure de recordar donde guardar las dos llaves ya que son importantes.

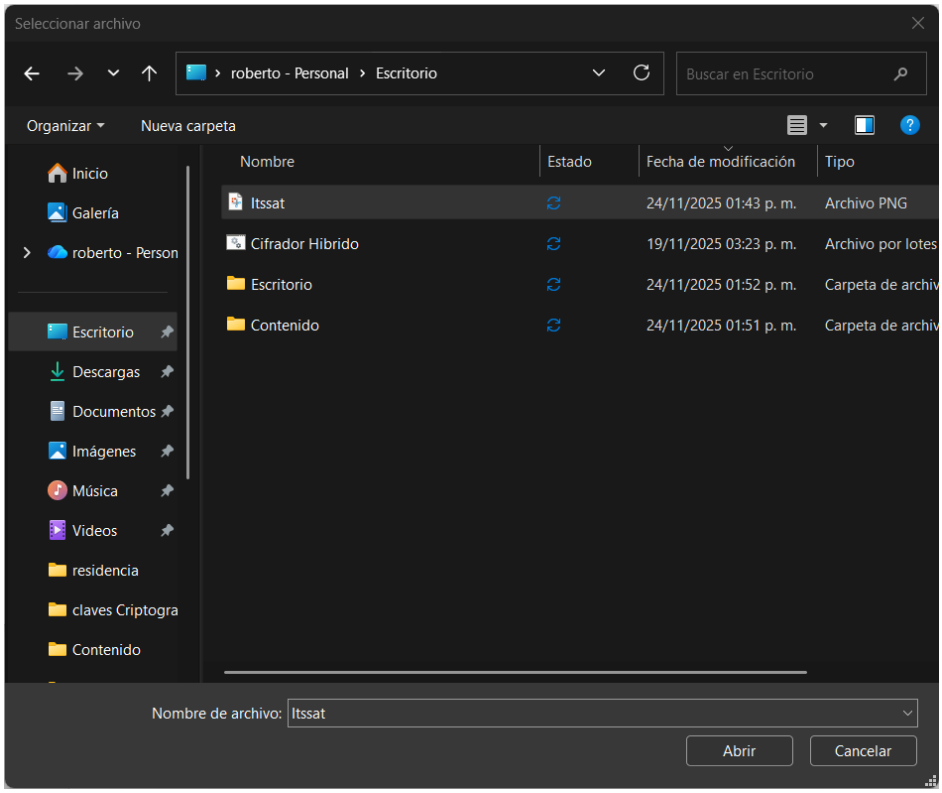
Ubicamos el archivo que vamos a cifrar en este caso será la siguiente imagen que se llama Itssat.

Ahora seguiremos en orden el paso dos.

En el primer apartado vamos a seleccionar la llave publica ósea la primera que nos genero y la vamos a seleccionar.

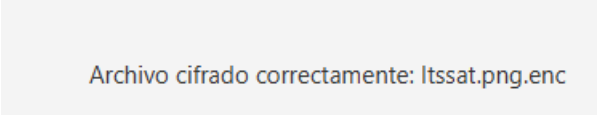


Ahora seleccionamos el archivo que vamos a cifrar como ya sabemos será la imagen.

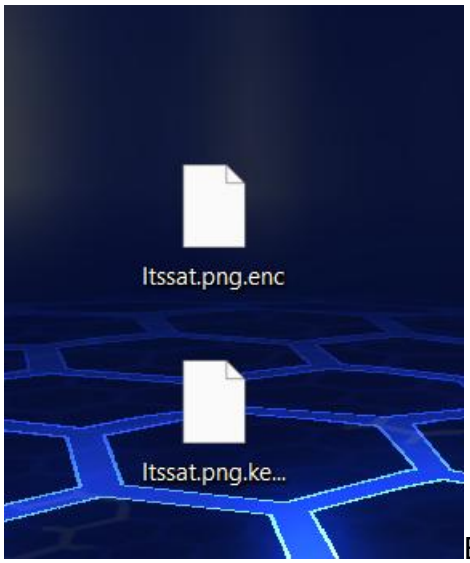


Tendremos que tener nuestra interfaz de la siguiente manera.

Si todo es correcto le daremos en Cifrar Archivo



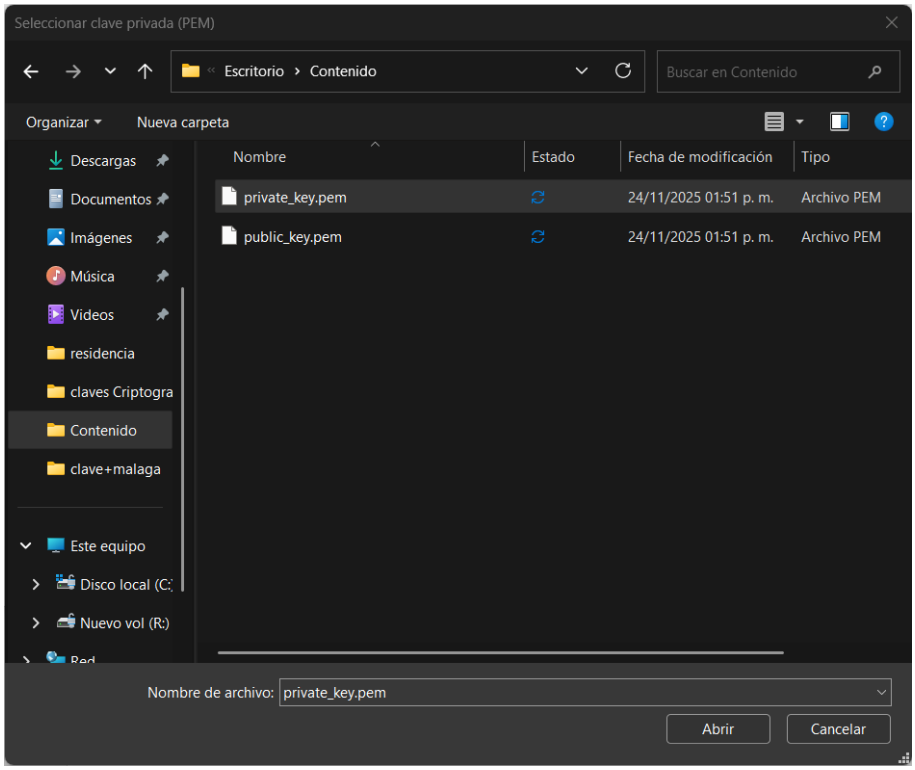
Cifrado Exitoso

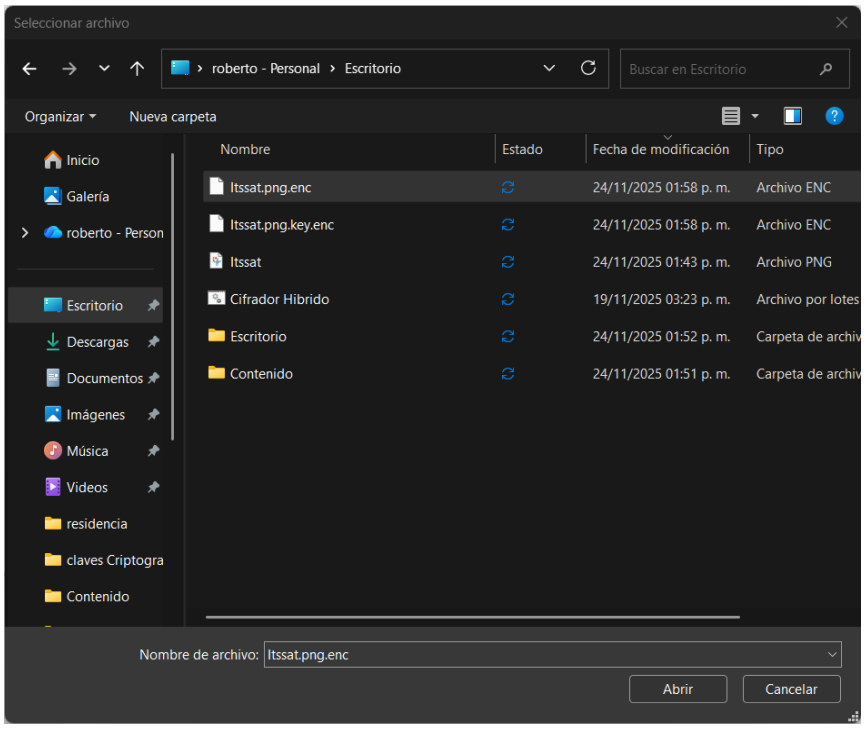


En nuestro escritorio se generarán dos archivos uno el archivo cifrado y el otro la llave.

Ahora para desencriptar el archivo usaremos la opción 3

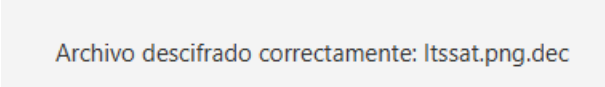
Realizaremos el mismo proceso solo que ahora será la llave privada y seleccionaremos el archivo encriptado.





Si todo es correcto debemos tenerlo de la siguiente manera.

Ahora le damos en Descifrar Archivo



Descifrado Exitoso.

Y como verán a regresado el archivo en su forma original.

7. CARACTERISTICAS

Puede cifrar absolutamente cualquier archivo:

Documentos: Word (.docx), PDF (.pdf), Excel (.xlsx), Texto (.txt).

Imágenes: Fotos (.jpg, .png), Diseños (.psd).

Multimedia: Videos (.mp4, .mkv), Audio (.mp3).

Ejecutables y Binarios: Programas (.exe), Librerías (.dll), Zips (.zip, .rar).

Limitaciones:

Carpetas completas: El FileChooser está diseñado para seleccionar archivos (File), no directorios. Para cifrar una carpeta, primero se tiene que comprimir en un .zip y luego cifrar ese .zip.

Cifrado en lote: La interfaz actual (selectedFileEncrypt) almacena solo un archivo en memoria a la vez para procesarlo.

8. CONCLUSIONES

El desarrollo del cifrador híbrido RSA–AES permitió integrar en un solo sistema las ventajas de la criptografía simétrica y asimétrica, logrando un equilibrio entre seguridad, eficiencia y facilidad de uso. AES proporcionó un cifrado rápido y adecuado para manejar archivos de cualquier tipo y tamaño, mientras que RSA aseguró la protección de la clave simétrica mediante un esquema moderno y confiable.

La arquitectura implementada, junto con el registro automático de operaciones en la base de datos, ofrece trazabilidad, control y organización del historial de cifrado y descifrado, lo cual incrementa la confiabilidad del sistema. Además, la interfaz gráfica desarrollada en JavaFX facilita la operación del software incluso para usuarios sin conocimientos avanzados en criptografía.

Gracias a estas características, el sistema cumple con los estándares actuales de seguridad informática y demuestra su utilidad como herramienta para proteger información sensible. Su capacidad para cifrar prácticamente cualquier archivo individual y su funcionamiento estable en diferentes pruebas confirman que la solución es funcional, escalable y aplicable en escenarios reales donde la protección de datos es esencial.