

# Manual de Procedimientos: Esquema de Secreto Compartido de Shamir

Lopez Molina Andrés Daniel      Orea Romero Laura Victoria

25 Noviembre 2024

## Índice

<b>1. Introducción</b>	<b>3</b>
1.1. Lineamientos Legales y Regulatorios . . . . .	3
<b>2. Estructura del Proyecto</b>	<b>6</b>
2.1. Objetivo del Proyecto . . . . .	6
2.2. Modalidades de Funcionamiento . . . . .	6
2.3. Diagrama de flujo: . . . . .	7
<b>3. Destinatarios del Proyecto</b>	<b>8</b>
<b>4. Roles y Responsabilidades</b>	<b>9</b>
<b>5. Información del Software</b>	<b>11</b>
5.1. Tipo de Software . . . . .	11
5.2. Requerimientos del Sistema . . . . .	11
5.3. Instalación y Configuración del Entorno . . . . .	12
5.4. Requerimientos para el Usuario Final . . . . .	12
5.5. Cifrado de Archivos con AES . . . . .	13
5.6. Descifrado de Archivos con AES . . . . .	13
5.7. Generación de Fragmentos para el Secreto . . . . .	13
5.8. Reconstrucción del Secreto . . . . .	14
5.9. Proceso de Cifrado Completo . . . . .	14
5.10. Proceso de Descifrado Completo . . . . .	14
<b>6. Explicación Técnica: Alta Entropía en SHA-256</b>	<b>15</b>

<b>7. Código de Implementación</b>	<b>15</b>
7.1. Cifrado . . . . .	15
7.2. Descifrado . . . . .	16
7.3. Generación de Fragmentos . . . . .	17
7.4. Interfaz Gráfica . . . . .	17
7.5. Procedimientos de Mantenimiento . . . . .	17

# 1. Introducción

La **esteganografía** es una disciplina que nos permite ocultar información dentro de otros datos de tal forma que pase desapercibida para quienes no tienen el conocimiento o acceso al mensaje original. En este proyecto, se implementará un sistema de esteganografía utilizando el método de **Least Significant Bit (LSB)** en imágenes, el cual permite esconder datos alterando el bit menos significativo de cada byte de la imagen, produciendo cambios imperceptibles en la imagen resultante.

## 1.1. Lineamientos Legales y Regulatorios

La implementación del SSS están regulados en varios países y pueden estar sujetas a lineamientos específicos en cuanto a privacidad y seguridad de la información

- Marco Regulatorio en México:
  - Ley Federal de Protección de Datos Personales en Posesión de los Particulares (LFPDPPP)
    1. Aplicación: El uso de SSS debe garantizar la protección de datos personales durante el proceso de cifrado, almacenamiento y reconstrucción.
    2. Requisitos: Asegurar que los datos confidenciales estén protegidos mediante medidas de seguridad administrativas, técnicas y físicas y garantizar que los fragmentos no puedan ser utilizados de manera independiente para comprometer la confidencialidad del secreto.
    3. Obligación: Notificar a los titulares de los datos sobre las medidas implementadas para proteger la información.
  - Ley de Transparencia y Acceso a la Información Pública
    1. En el caso de entidades públicas, el manejo de datos sensibles mediante SSS debe cumplir con los principios de seguridad y confidencialidad, asegurando que los fragmentos generados sean inaccesibles para terceros no autorizados.
  - Normas Oficiales Mexicanas (NOM)
    1. NOM-151-SCFI-2016: Regulación relacionada con la conservación y seguridad de mensajes electrónicos, aplicable al almacenamiento de fragmentos en medios digitales.
    2. NOM-019-SCFI-2011: Supervisión de sistemas de información y tecnología, relevante para la implementación técnica del esquema.

- Normativas Internacionales
  - Regulaciones de Protección de Datos
    - General Data Protection Regulation (GDPR) - Unión Europea:
      1. SSS debe cumplir con principios como el minimización de datos, garantizando que los fragmentos almacenados sean estrictamente los necesarios.
      2. La encriptación y el manejo de claves mediante SSS se consideran medidas de seguridad adecuadas para el cumplimiento del artículo 32 del GDPR.
    - California Consumer Privacy Act (CCPA) - Estados Unidos:
      1. Implementar SSS en el manejo de información confidencial de consumidores califica como medida de protección ante accesos no autorizados.
  - Requisitos Criptográficos
    - ISO/IEC 27001: El uso de SSS debe alinearse con los estándares internacionales de gestión de seguridad de la información.
    - NIST SP 800-57 (Estados Unidos): Especificaciones para la gestión de claves criptográficas, aplicables al uso y almacenamiento de fragmentos generados por SSS.
  - Regulaciones Financieras
    - Basel III (Banca Internacional): Para entidades financieras que empleen SSS para proteger activos o información sensible, el esquema debe integrarse dentro de un marco robusto de gestión de riesgos operativos.
    - Ley USA PATRIOT Act (EE.UU.): En aplicaciones relacionadas con el resguardo de datos de entidades financieras, el sistema debe prevenir accesos no autorizados y garantizar trazabilidad en la reconstrucción de secretos.
- Recomendaciones Operativas para el Cumplimiento
  - Seguridad y Control de Acceso
    - Establecer controles de acceso estrictos para los fragmentos generados, asegurando que únicamente personal autorizado tenga acceso a ellos.
    - Registrar de forma auditable cada acceso, modificación o eliminación de los fragmentos.

- Custodia de Fragmentos
  - Implementar políticas claras sobre el almacenamiento de los fragmentos:
    1. Fragmentos distribuidos geográficamente para mitigar riesgos de pérdida total.
    2. Uso de almacenamiento cifrado en medios digitales o físicos
- Uso Responsable
  - No almacenar fragmentos en servicios no autorizados o inseguros.
  - Realizar auditorías periódicas para verificar la integridad y disponibilidad de los fragmentos.
- Consentimiento y Transparencia
  - Informar a los usuarios finales o partes interesadas sobre la metodología utilizada para proteger su información.
  - Asegurar el consentimiento explícito para el uso de SSS en datos personales o sensibles.
- Sanciones por Incumplimiento México
  - México
    - La LFPDPPP impone sanciones de hasta 320,000 días de salario mínimo por el mal manejo de datos personales.
    - Incumplir con las normas de seguridad de información podría derivar en responsabilidad administrativa o penal.
  - Internacional
    - En el ámbito del GDPR, las sanciones pueden alcanzar el 4 % de la facturación global de la organización por violaciones graves.
    - La CCPA impone multas de hasta \$7,500 USD por cada violación de datos.
- Evaluación y Actualización del Esquema
  - Realizar revisiones regulares de las implementaciones técnicas y operativas del esquema SSS para adaptarse a cambios normativos.
  - Monitorear las actualizaciones de regulaciones locales e internacionales relevantes para garantizar la continua conformidad.

## 2. Estructura del Proyecto

### 2.1. Objetivo del Proyecto

Implementar un sistema de compartición de claves basado en el algoritmo de Shamir's Secret Sharing (SSS) que permita dividir y distribuir un secreto de forma segura entre múltiples partes, garantizando su recuperación mediante un número mínimo requerido de partes y ofreciendo una solución práctica y confiable para la gestión y protección de claves criptográficas en entornos colaborativos.

### 2.2. Modalidades de Funcionamiento

#### ■ Cifrado

1. Abre la aplicación y selecciona "Cifrar Archivo".
2. Proporciona los siguientes elementos:
  - Archivo Original: Selecciona el archivo que deseas cifrar.
  - Archivo Cifrado: Define el nombre y la ubicación del archivo cifrado resultante.
  - Archivo de Fragmentos: Define el nombre y la ubicación para guardar los fragmentos generados.
3. Especifica:
  - "n": Número total de fragmentos.
  - "t": Número mínimo de fragmentos requeridos para descifrar.
4. Ingresa una contraseña segura cuando se solicite.
5. La aplicación generará el archivo cifrado y sus fragmentos.

#### ■ Descifrado

1. Abre la aplicación y selecciona "Descifrar Archivo".
2. Proporciona los siguientes elementos:
  - Archivo de Fragmentos: Selecciona el archivo que contiene los fragmentos necesarios.
  - Archivo Cifrado: Selecciona el archivo previamente cifrado.
  - Archivo Descifrado: Define el nombre y la ubicación del archivo reconstruido.
3. La aplicación reconstruirá el archivo original y lo almacenará en la ubicación especificada.

## 2.3. Diagrama de flujo:

### 1. Cifrado:

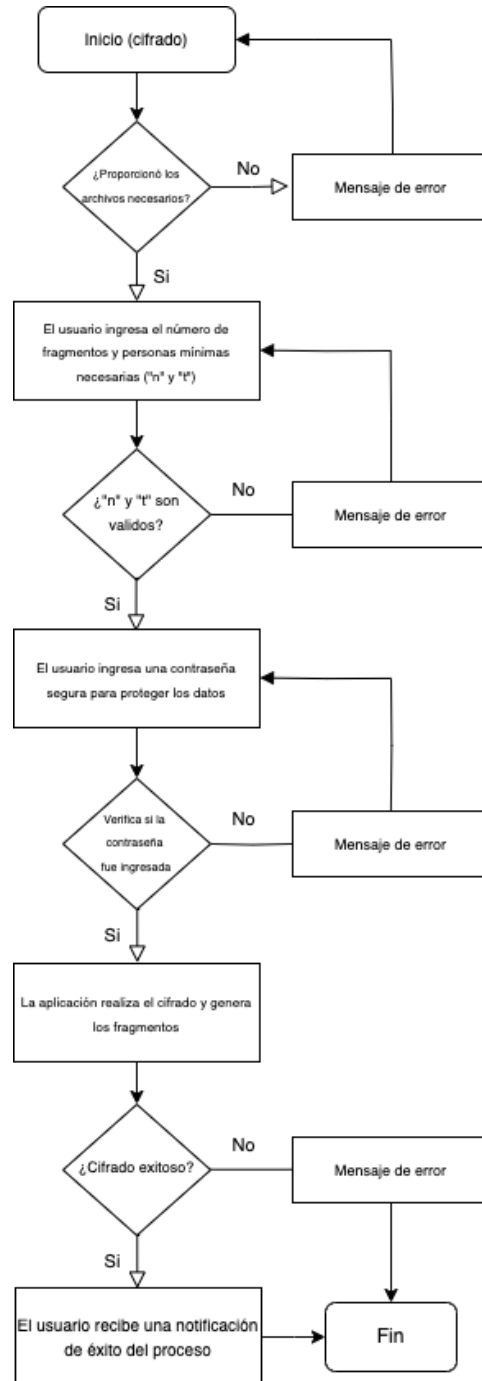


Figura 1: Diagrama de flujo del cifrado

## 2. Descifrado:

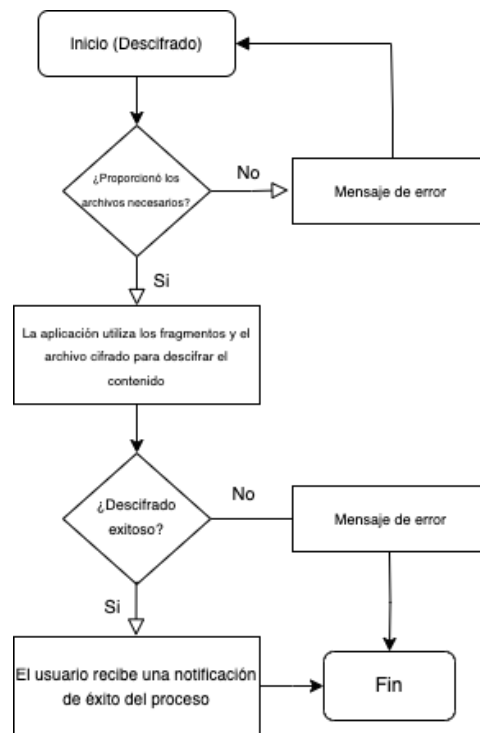


Figura 2: Diagrama de flujo del descifrado

## 3. Destinatarios del Proyecto

El proyecto “*Esquema de Secreto Compartido de Shamir*” está dirigido a:

- **Organizaciones con información sensible:** Empresas, gobiernos o instituciones que necesitan compartir claves de cifrado o datos críticos y/o sensibles entre múltiples responsables, garantizando que solo un grupo mínimo pueda acceder a la información
- **Administradores de sistemas y seguridad informática:** Profesionales encargados de gestionar claves que estén interesados en mejorar la resiliencia y confiabilidad de sus sistemas de seguridad.
- **Entornos de respaldo y recuperación de datos:** Proyectos que buscan asegurar datos sensibles o claves, dividiendo las responsabilidades entre varias personas, para evitar pérdida de información por fallas individuales.



- **Grupos colaborativos con alto nivel de confianza:** Equipos de liderazgo, juntas directivas, o grupos de inversión que deseen proteger información compartida sin depender de un único individuo.

## 4. Roles y Responsabilidades

Para este proyecto, se consideran los siguientes roles principales:

### Jefe de Proyecto

El Jefe de Proyecto (Andrés Daniel López Molina) es el responsable de supervisar y coordinar todas las actividades del equipo para garantizar que el proyecto se complete a tiempo, dentro del presupuesto y cumpliendo con los requisitos establecidos. Sus responsabilidades incluyen:

- Planificación y gestión de cronograma.
- Supervisión del presupuesto del proyecto.
- Coordinación entre el equipo de desarrollo y los stakeholders.
- Gestión de riesgos.
- Toma de decisiones y resolución de conflictos.

### Desarrollador de Software

Los desarrolladores de software (Andrés Daniel López Molina y Laura Victoria Orea Romero) son responsables de escribir, probar y mantener el código del proyecto. En el contexto de un proyecto de desarrollo web, este rol puede incluir:

- Desarrollo del lado del servidor (back-end) y del cliente (front-end).
- Implementación de la lógica de negocio.
- Creación de bases de datos y gestión de datos.
- Pruebas unitarias y de integración.
- Depuración de código.

## **Desarrollador Back-End**

El Desarrollador Back-End (Laura Victoria Orea Romero) es responsable de la lógica del lado del servidor y de la gestión de las bases de datos. Sus responsabilidades incluyen:

- Creación de servidores, bases de datos y API.
- Gestión de la seguridad y la autenticación de usuarios.
- Optimización del rendimiento del sitio web.
- Integración de servicios y herramientas de terceros.

## **Tester o QA (Aseguramiento de la Calidad)**

El Tester o QA (Andrés Daniel López Molina) se encarga de garantizar que el software desarrollado funcione correctamente y cumpla con los requisitos establecidos. Sus responsabilidades incluyen:

- Pruebas de funcionalidad, rendimiento y seguridad.
- Identificación y reporte de errores y fallos.
- Creación de pruebas automatizadas.
- Validación de la calidad del producto antes de su lanzamiento.

## **Administrador de Sistemas**

El Administrador de Sistemas (Laura Victoria Orea Romero) se encarga de la infraestructura técnica del proyecto. Sus responsabilidades incluyen:

- Gestión y mantenimiento de servidores web y bases de datos.
- Implementación de políticas de seguridad.
- Monitoreo del rendimiento del sistema.
- Gestión de backups y recuperación ante desastres.

## Product Owner

El Product Owner (María Ximena Lezama Hernández) es responsable de definir los requisitos del producto y asegurar que el desarrollo cumpla con las expectativas del cliente. Sus responsabilidades incluyen:

- Definir las características y funcionalidades del producto.
- Priorización de tareas y funcionalidades en el backlog.
- Actuar como intermediario entre los stakeholders y el equipo de desarrollo.
- Asegurar que el producto final cumpla con los requisitos del cliente.

**Dependencias:** Nosotros como programadores del sistema solucionamos errores técnicos e implementamos mejoras.

## 5. Información del Software

### 5.1. Tipo de Software

El software que desarrollamos implementa un esquema de secreto compartido basado en el algoritmo de Shamir. Está diseñado para cifrar y descifrar archivos de manera segura, utilizando criptografía simétrica (AES) y conceptos de polinomios de interpolación. A continuación, se describen sus características principales:

- **Funcionalidad Principal:** Cifrado y descifrado de documentos confidenciales con distribución de fragmentos clave.
- **Categoría:** Aplicación de escritorio de criptografía y gestión de claves.
- **Tipo:** Interfaz gráfica e interacción mediante línea de comandos.

### 5.2. Requerimientos del Sistema

Para utilizar el software de la aplicación, de preferencia se deben cumplir los siguientes requisitos para su adecuado :

#### Hardware

- Procesador: CPU de al menos 2 GHz.
- Memoria RAM: 4 GB (recomendado 8 GB para mayor fluidez).
- Espacio en disco: Al menos 50 MB para instalar dependencias y almacenar archivos cifrados.

## Software

- Sistema Operativo: Windows 10/11, macOS 10.15+ o Linux (Ubuntu 20.04 o superior).
- Lenguaje de programación utilizado: Python 3.8
- Bibliotecas requeridas:
  - `pycryptodome` para manejo de cifrado AES.
  - `tkinter` para la interfaz gráfica.
  - `hashlib` para generación de claves seguras (SHA-256).

### 5.3. Instalación y Configuración del Entorno

El software requiere la instalación de ciertas dependencias. A continuación, se describen los pasos en caso de no contar previamente con estas:

1. **Instalación de Python:** Descarga Python desde <https://www.python.org/>. Durante la instalación, asegúrate de activar la opción *“Add Python to PATH”*.
2. **Instalación de Bibliotecas:** Se deberá abrir una terminal y ejecutar los siguientes comandos en caso de que no se cuente con las paqueterías:

```
pip install pycryptodome
pip install tk
```

3. **Ejecución del Software:** Deberás descargar el código fuente y ejecutar el archivo principal desde la terminal:

```
python interfaz.py
python3 interfaz.py
```

### 5.4. Requerimientos para el Usuario Final

Para usar el software, se espera que de preferencia el usuario cuente con:

- Conocimientos básicos en manejo de archivos.
- Habilidad para gestionar las contraseñas y fragmentos generados.

## 5.5. Cifrado de Archivos con AES

**Función:** `cifrar_documento`

- **Descripción:** Esta función cifra un archivo utilizando el algoritmo AES en modo EAX.
- **Entrada:**
  - Archivo original.
  - Clave de cifrado de 256 bits (en formato hexadecimal).
- **Salida:** Archivo cifrado que contiene el *nonce*, etiqueta de autenticidad (*tag*) y datos cifrados.

## 5.6. Descifrado de Archivos con AES

**Función:** `descifrar_documento`

- **Descripción:** Esta función descifra un archivo previamente cifrado con AES.
- **Entrada:**
  - Archivo cifrado.
  - Clave de 256 bits (en formato hexadecimal).
- **Salida:** Archivo descifrado con el contenido original.

## 5.7. Generación de Fragmentos para el Secreto

**Función:** `generar_polinomio`

- **Descripción:** Divide el secreto (clave de cifrado) en  $n$  fragmentos mediante un polinomio de grado  $t - 1$ .
- **Entrada:**
  - Secreto ( $k$ ).
  - Número de fragmentos ( $n$ ).
  - Número mínimo para reconstrucción ( $t$ ).
- **Salida:** Lista de pares  $(x, P(x))$  que representan los fragmentos.

## 5.8. Reconstrucción del Secreto

**Función:** `reconstruir_secreto`

- **Descripción:** Recupera el secreto utilizando al menos  $t$  fragmentos mediante interpolación de Lagrange.
- **Entrada:** Lista de pares  $(x, y)$  de fragmentos disponibles.
- **Salida:** Término independiente del polinomio (secreto  $k$ ).

## 5.9. Proceso de Cifrado Completo

**Función:** `cifrar_proceso`

- **Descripción:** Integra las etapas de cifrado:
  1. Genera una clave SHA-256 a partir de la contraseña del usuario.
  2. Divide la clave en  $n$  fragmentos utilizando un polinomio de grado  $t - 1$ .
  3. Cifra el archivo con AES y guarda los fragmentos en un archivo separado.
- **Entrada:** Contraseña,  $n$ ,  $t$ , archivo original.
- **Salida:** Archivo cifrado y archivo con fragmentos.

## 5.10. Proceso de Descifrado Completo

**Función:** `descifrar_proceso`

- **Descripción:** Integra las etapas de descifrado:
  1. Reconstruye la clave usando los fragmentos proporcionados.
  2. Descifra el archivo con la clave reconstruida.
- **Entrada:** Fragmentos y archivo cifrado.
- **Salida:** Archivo descifrado.

## 6. Explicación Técnica: Alta Entropía en SHA-256

La alta entropía de la clave generada mediante SHA-256 asegura un alto nivel de aleatoriedad y resistencia a ataques. La clave:

- Tiene una longitud fija de 256 bits.
- Es única para cada contraseña, gracias a la uniformidad del hash.
- Garantiza un espacio de búsqueda tan amplio que es prácticamente imposible de predecir mediante fuerza bruta.

## 7. Código de Implementación

### 7.1. Cifrado

---

```
1 from Crypto.Cipher import AES
2
3
4 NONCE_SIZE = 16
5 TAG_SIZE = 16
6
7 def cifrar_documento(archivo_claro, archivo_cifrado, clave):
8
9     if len(clave) != 64:
10         raise ValueError("La clave debe ser de 256 bits (64 caracteres
11                               hexadecimales).")
12
13     clave_bytes = bytes.fromhex(clave)
14     cipher = AES.new(clave_bytes, AES.MODE_EAX)
15     nonce = cipher.nonce
16
17     try:
18         with open(archivo_claro, 'rb') as archivo:
19             datos = archivo.read()
20     except FileNotFoundError:
21         raise FileNotFoundError(f"No se encontro el archivo original:
22                                   {archivo_claro}")
23
24     datos_cifrados, tag = cipher.encrypt_and_digest(datos)
```

```
24 with open(archivo_cifrado, 'wb') as archivo:
25     archivo.write(nonce)
26     archivo.write(tag)
27     archivo.write(datos_cifrados)
28
29 print(f"Archivo cifrado correctamente: {archivo_cifrado}")
```

---

## 7.2. Descifrado

---

```
1 [language=Python]
2 def descifrar_documento(archivo_cifrado, archivo_descifrado, clave):
3
4     if len(clave) != 64:
5         raise ValueError("La clave debe ser de 256 bits (64 caracteres
6             hexadecimales).")
7
8     clave_bytes = bytes.fromhex(clave)
9
10    try:
11        with open(archivo_cifrado, 'rb') as f:
12            nonce = f.read(NONCE_SIZE)
13            tag = f.read(TAG_SIZE)
14            datos_cifrados = f.read()
15    except FileNotFoundError:
16        raise FileNotFoundError(f"No se encontro el archivo cifrado:
17            {archivo_cifrado}")
18    except Exception as e:
19        raise Exception(f"Error al leer el archivo cifrado: {str(e)}")
20
21    try:
22        cipher = AES.new(clave_bytes, AES.MODE_EAX, nonce=nonce)
23        datos_descifrados = cipher.decrypt_and_verify(datos_cifrados,
24            tag)
25    except ValueError:
26        raise ValueError("Error al descifrar: La clave es incorrecta o
27            los datos estan corruptos.")
28
29    with open(archivo_descifrado, 'wb') as f:
30        f.write(datos_descifrados)
```



```
28     print(f"Archivo descifrado correctamente. Guardado en:
        {archivo_descifrado}")
```

---

### 7.3. Generación de Fragmentos

---

```
1  import random
2  from functools import reduce
3
4  MODULO = 2**256 - 189
5
6  def generar_polinomio(k, n, t):
7      coeficientes = [k] + [random.randint(1, MODULO - 1) for _ in range(t
8          - 1)]
9      puntos = [(x, evaluar_polinomio(coeficientes, x)) for x in range(1,
10         n + 1)]
11     return puntos
```

---

### 7.4. Interfaz Gráfica

- Pantallas principales:

1. Pantalla de selección de modo: *Cifrar* o *Descifrar*.
2. Pantalla de selección de archivos y parámetros (número de fragmentos  $n$ , mínimo necesario  $t$ ).

- Aspectos visuales:

- Colores predominantes:

- Fondo: #252525 (tono oscuro para resalte y facilidad visual).
- Botones: #87cefa (azul suave) y #32CD32 (verde para acciones clave).

- Tipografía:

- Títulos: Helvetica, tamaño 16-18 px.
- Texto en botones: Arial, tamaño 12 px.

### 7.5. Procedimientos de Mantenimiento

- Actualización de dependencias: Ejecuta regularmente:

```
pip install --upgrade pycryptodome
```

- Revisión de seguridad: Como recomendación deberás mantener actualizado el sistema operativo para asegurar soporte de cifrado.