

Sistema de Cifrado de Documentos con Vigenère y Fernet

Autor - David Ricardo Ordoñez Mora

Sistema de Cifrado y Descifrado de Archivos .txt, .pdf y .docx utilizando Algoritmos Vigenère y Fernet con Interfaz Gráfica en Python

Objetivos

Objetivo General

Desarrollar un sistema de cifrado y descifrado de documentos que soporte los formatos .txt, .pdf y .docx, utilizando los algoritmos de Vigenère y Fernet, con interfaz de usuario gráfica y registro de errores.

Objetivos Específicos

- Implementar el algoritmo clásico de Vigenère para cifrar y descifrar texto alfabético.
- Aplicar el cifrado simétrico Fernet para garantizar confidencialidad en archivos.
- Leer y escribir archivos en formatos .txt, .pdf y .docx.
- Desarrollar una interfaz gráfica con Tkinter para facilitar el uso del sistema.
- Registrar errores del sistema en un archivo de log para su depuración.

¿Qué es el cifrado?

El **cifrado** es una técnica que transforma un mensaje para que solo lo puedan leer personas autorizadas. Imagina que escribes una carta y la encierras en una caja con candado: Solo quien tenga la llave puede leerla. Así funciona el cifrado: **protege la información**, haciendo que un archivo se vea ilegible si alguien sin permiso lo abre.

“Piensa en el cifrado como poner tu información en una caja fuerte. Solo quien tenga la **clave correcta** podrá abrirla” Es una de las herramientas principales de la **ciberseguridad**, usada para proteger:

- Mensajes personales
- Archivos confidenciales
- Documentos en la nube
- Contraseñas y más

Cifrado de Vigenere

Es un método de cifrado por sustitución poli alfabética. Usa una palabra clave para determinar cómo cambiar cada letra del texto original.

- Ejemplo: Si la clave es SOL y el texto es HOLA, cada letra del texto se desplaza en el alfabeto según la letra correspondiente de la clave.

¿Cómo lo implementa Python?

En tu proyecto, el cifrado Vigenère:

- Recorre cada letra del texto.
- Usa funciones como `ord()` y `chr()` para convertir letras en números y viceversa.
- Aplica una fórmula matemática que toma en cuenta la clave y la posición de cada letra.

Este método funciona con archivos `.txt`, `.pdf` y `.docx`. Python extrae el texto del archivo, lo cifra letra por letra, y luego lo vuelve a guardar.

¿Para qué lo usamos en el proyecto?

- Para demostrar cómo se construyen los algoritmos de cifrado desde cero.
- Para enseñar a los usuarios cómo una clave afecta el contenido del mensaje.
- Para fines didácticos, mostrando que incluso métodos antiguos pueden proteger información básica.
-

Cifrado de Fernet

Fernet es parte del módulo `cryptography` de Python y representa un algoritmo de cifrado simétrico seguro. Usa una clave secreta aleatoria para cifrar y descifrar datos.

“Es como tener una llave maestra que convierte archivos legibles en datos cifrados imposibles de leer sin la clave”.

¿Cómo lo implementa Python?

En tu proyecto:

- Se genera una clave Fernet con `Fernet.generate_key()` y se guarda.
- El archivo se lee como texto.
- El texto se convierte a bytes (`.encode('utf-8')`).
- Se cifra con `fernet.encrypt()`.
- Se guarda como un archivo `.cif` (binario cifrado).

- Luego se puede descifrar con `fernet.decrypt()` y reconstruir el archivo en su formato original (.txt, .pdf, .docx).

“Python garantiza que cada dato cifrado tenga integridad y autenticidad, protegiéndolo incluso de alteraciones”.

¿Para qué lo usamos en el proyecto?

- Para ofrecer una alternativa **segura y realista** de cifrado de archivos personales.
- Para practicar el uso de librerías profesionales de criptografía en Python.
- Para enseñar a proteger información de forma **automatizada** y reutilizable, incluso para quienes no saben programar.

¿Por qué usar ambos métodos en un mismo proyecto?

Método	Propósito en el proyecto
Vigenère	Enseñar la lógica básica del cifrado con letras y claves.
Fernet	Mostrar un ejemplo de cifrado seguro y aplicable en la vida real .

- Con esto, el usuario puede **ver la evolución del cifrado**: desde lo clásico y manual hasta lo moderno y automatizado.

¿Cómo facilita Python este proyecto?

- Permite leer y escribir distintos tipos de archivo fácilmente (txt, pdf, docx).
- Usa módulos como `cryptography`, `docx`, `reportlab`, y `PyPDF2`.
- Maneja errores con logging y archivos .log para registrar fallos.
- Ofrece una interfaz gráfica (tkinter) que **no requiere conocimientos de programación** para usarla.

Estructura del Proyecto

```

|—— __pycache__/
|—— claves/
|   |—— clave.key      # Clave de Fernet generada
|—— funciones/
|   |—— __init__.py
|   |—— fernet.py      # Lógica Fernet.
|   |—— vigenere.py    # Lógica Vigenere.
```

	logs/	
		error.log # Registro de errores.
	interfaz.py	# Interfaz gráfica con Tkinter.
	main.py	# Ejecuta cifrado y descifrado con Vigenère.
	mainFernet.py	# Ejecuta cifrado y descifrado con Fernet.
	prueba_gui.py	# Prueba interfaz gráfica.
	requerimientos.txt	# Dependencias.
	readme.md	#

Descripción de Archivos y Módulos

funciones/vigenere.py

Contiene funciones para:

- leerArchivo (): lee archivos .txt, .pdf y .docx.
- guardarArchivo (): guarda texto en dichos formatos (PDF usa ReportLab).
- cifrarVigenere (): implementa el algoritmo clásico de Vigenère.
- descifrarVigenere (): realiza el descifrado correspondiente.

funciones/fernet.py

Contiene funciones para:

- generarClave (): genera y guarda una clave Fernet.
- cargarClave (): lee una clave Fernet desde un archivo.
- leerContenido (): lee el contenido según el tipo de archivo.
- escribirContenido (): guarda el contenido descifrado en su formato original.
- cifrarArchivo (): cifra un archivo con Fernet.
- descifrarArchivo (): descifra y reconstruye el archivo.

interfaz.py

- Interfaz gráfica desarrollada con Tkinter.
- Permite seleccionar archivo, algoritmo y tipo de operación.
- Ejecuta procesos y muestra resultados.

main.py

- Ejecuta directamente el cifrado y descifrado con Vigenère sobre un archivo.

mainFernet.py

- Ejecuta directamente el cifrado y descifrado con Fernet sobre un archivo.

logs/error.log

- Registro automático de errores y excepciones para seguimiento de fallos.

Requisitos

Incluidos en requerimientos.txt:

- PyPDF2
- reportlab
- python-docx
- tk
- cryptography

Instalación:

- `pip install -r requerimientos.txt`

Ejecución - Vía consola:

- `python main.py`
- `python mainFernet.py`

Vía interfaz gráfica:

- `python prueba_gui.py`

Notas Adicionales

- El sistema guarda por defecto los archivos cifrados y descifrados en la misma ubicación del archivo original, con sufijos `_Vigenere`, `_Vigenere_Descifrado`, `_Fernet`, `_Descifrado`, según corresponda.
- Para cifrado Fernet, el archivo `claves/clave.key` debe existir o ser generado antes.