UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ciencias

Shamir's Secret Sharing Scheme

Modelado y programación

Autores:
Valencia Cruz Jonathan Josué
318280211
Aguirre Muñoz Leonardo
318017686

25 de enero de 2022

1. Definición del problema

A partir de la implementación del Esquema de Shamir (Shamir's Secret Sharing Schemes, o SSSS por sus siglas en inglés) y de Advanced Encryption Standard (AES), se solicita encriptar y desencriptar un archivo dado.

El programa deberá funcionar a partir de dos opciones a petición del usuario:

Encriptar. Se debe proporcionar, en la línea de llamada:

1. Ingresar el número de claves a generar.
2. Ingresar el número de claves requeridas para desencriptar el archivo.
3. Mostrar en pantalla las claves generadas en el proceso.
4. Solicitar el nombre y ruta del archivo a encriptar.
5. Solicitar el nombre y ruta del archivo encriptado

Desencriptar. Se debe proporcionar, en la línea de llamada:

1. Ingresar el número de claves a ingresar.
2. Solicitar las claves requeridas.
3. Solicitar el nombre y ruta del archivo encriptado.
4. Solicitar el nombre y ruta del archivo desencriptado.

2. Análisis del problema

Para la resolución del problema, planteamos realizar diferentes clases. a continuación se presenta cada una de las clases adjunto a su descripción:

- Clase SSSS: Esta clase se encargará del encriptamiento dado un archivo, esto se realizará a partir de una llave generada de manera aleatoria y, a partir de la llave, se generará n número de claves obtenidas a partir del Esquema de Shamir que posteriormente se mostrarán en pantalla. A partir de la llave, se encriptará el archivo por AES y se guardará a petición del usuario.
- Clase desencriptar: Se especializará en el desencriptamiento dado un archivo, específicamente se realizará obteniendo las claves requeridas para el desencriptamiento del archivo además del archivo en si, posteriormente verificará su autenticidad y, en caso de ser afirmativo, se almacenará el archivo desencriptado a petición del usuario.
- Clase Main: Funcionará como clase principal y servirá para la creación de la interfaz de usuario e implementación de las demás clases. Será el archivo a ejecutar para el uso del programa.

3. Selección de la mejor alternativa

En este caso, trabajaremos con los siguientes parámetros para la realización del proyecto:

- Se utilizará el formato .txt para el encriptamiento/desencriptamiento de archivos dado que es versatil para el guardado de información, indicaciones del proyecto y por su fácil manejo de importación/exportación de datos.
- Se usará la biblioteca PyCryptodome integrada en Python por su fácil uso y porque facilitará en gran medida la realización del proyecto.
- Para la encriptación de archivos se usarán el Esquema de Shamir y AES, esto por indicaciones del proyecto además ser formas fiables de encriptación
- El lenguaje de programación a utilizar será Python, esto debido a su gran repertorio de bibliotecas para el procesamiento de texto y criptografía, creación de interfaces gráficas de usuario y el fácil entendimiento de su sintaxis, lo que facilita en gran medida la legibilidad del programa.

4. Diagrama de flujo o pseudocódigo

```
#Clase que nos permitirá encriptar archivos
clase ocultar:
    #Función que genera la llave para encriptar un archivo
    obtener_llave():
        #Se produce un número de 16 bits al azar
        llave= obtenerllave(16)
        #Devuelve la clave generada
        regresa key

    #Generadora de claves por Esquema de Shamir a partir de una llave
    obtenerpiezas(llave)
        #Se le solicita al usuario el numero de fragmentos para la llave
        Repetir
            fragmentos_solicitados=escribir_numero_entero
            ("Ingresa el numero total de claves a generar")
            #condicion numero<0:
            error("Número no válido")
        Hasta que numero>0:

        #Pregunta al usuario cuantas claves son necesarias para desencriptar el archivo
        Repetir
            piezas_requeridas=escribir_numero_entero("Ingresa el numero total de claves a generar")
            #condicion piezas_requeridas<0 ó piezas_requeridas> fragmentos_solicitados:
            error("Número no válido")
        Hasta que (piezas_requeridas<0 ó piezas_requeridas>fragmentos_solicitados)

        #Genera las claves usando el esquema mencionado.
        fragmentos= Shamir.separar(piezas_requeridas,fragmentos_solicitados,llave)
        #Regresa los fragmentos generados de la llave
        regresar fragmentos

    #Imprime las claves en pantalla
    ruta=(tipo_archivo,nombre_archivo)
    guardar = (tipo_archivo,nombre_archivo)
    direccion = camino(ruta)
    nombre = Direccion(guardar)
    abrir(ruta,lectura) como entrada, abrir(nombre,escritura) como salida
    cifrar = Método_Encriptacion.nuevo(llave,Fragmentos.Encriptar)

    #Encripta un archivo dado
    encriptar(llave)

#Clase para descifrar un archivo encriptado dado
clase descifrar:
    #Función que almacena las claves ingresadas por los usuarios
    fragmentos=[] (lista)

    #Se le solicita al usuario el número de claves a utilizar
        Repetir
            personas=escribir_numero_entero
            ("Ingresa el numero de personas presentes")
            #condición personas<1:
            error("Número no válido")
        Hasta que personas<1:
    #Inicia una iteración para obtener las claves de las respectivas personas
    for x hasta personas
        #Se le solicita su clave
        cadena= ingresar("Escriba su clave")
```

```
    #Separa el número y la clave
    cadena= cadena.sin_espacios()
    tupla= cadena.quitar(':',una vez)
    clave= tupla.quitar('sobrante',una vez)

    #Almacena la clave y numero en una lista de tuplas siendo los fragmentos

    fragmentos.unir(entero(tupla),valorhexa(clave.quitar('\'')))
#Regresa la lista de claves dadas por los usuarios
regresar fragmentos

#Función que obtiene la lista de claves y las combina
obtenerclaves()

intenta:
    llave= EsquemaShamir.combinar(fragmentos)
esperar Error:
    ("Las claves son incorrectas")
#Devuelve la llave
regresa llave

#Desencripta el archivo dado por el usuario a partir de la llave recuperada
ruta=(tipo_archivo,nombre_archivo,"¿Qué archivo desea abrir?")
guardar=(tipo_archivo,nombre_archivo,"Generar_archivo_nuevo")

#Prepara la lectura del archivo y su proceso de descifrado
Abrir(ruta,"lectura") entrada:
#Obtiene las variables almacenadas en el archivo encriptado
piezas,tipo=leer(16 bits)
#Realiza una nueva llave
cifrar=Nueva(llave,Modo.EsquemaShamir,piezas)
#Si la llave es incorrecta, envía un mensaje en la pantalla
intenta:
    resultado=cifrado.desencriptado(lectura)
    cifrado.verificar(tipo)
    #Declara el proceso de escritura
    abrir(guardar,"escritura") salida
    # Si la llave es correcta, se crea el archivo desencriptado
    escribir(resultado)
esperar Error:
    ("Las claves son incorrectas o no son del archivo correspondiente")

#Clase para ejecutar el programa
clase menu:

#Funcion para encriptar
Encriptar():
#Llama la clase ocultar
Encripta=ocultar()
#Genera la llave
llave=Encripta.obtener_llave()
#Fragmenta la llave para encriptar el archivo
fragmentos=Encripta.obtener_fragmentos(llave)
#Muestra los fragmentos generados en forma de clave
Encripta.imprimir_fragmentos(fragmentos)
#Encripta el archivo solicitado
Encripta.encriptar(llave)

#Funcion para desencriptar
Desencriptar():
#Llama la clase descifrar
```

```
    Desencripta=descifrar()
    #Obtiene las claves necesarias para desencriptar el archivo
    fragmentos= Desencripta.obtener_fragmentos()
    #Obtiene una llave de las claves dadas
    llave=Desencripta(obtener_llave(fragnentos))
    #Desencripta el archivo
    Desencripta.desencriptar(llave)

    #Funcion para regresar al menú
    ("Volviendo al menú principal...")
    #Funcion para iniciar ejecutable
    Ejecutable():
       Repetir:
            intenta:
                opcion=entrada("\n---------------------------------
                ----------------------\nBienvenido a ENDEcryptApp,
                ¿Qué te gustaría hacer?\n\n1.-Encriptar un
                archivo\n\n2.-Desencriptar un
                archivo\n\n3.-Salir\n-----------------------------
                ------------------------\n"))
                if opcion==1:
                    Encriptar()
                    RegresarMenu()
                elif opcion==2:
                    Desencriptar()
                    RegresarMenu()
                elif opcion==3:
                    ("\n\n\nGracias por usar nuestra aplicación, vuelve pronto\n\n")
                    break
                elif opcion<1 or opcion>3:
                    error("")

            esperar Error:
                error()
Aplicacion=menu()
Aplicacion.Ejecutable()
```

5. Mantenimiento y proyecto a largo plazo

A lo largo de la realización de este proyecto se presentaron diversas complicaciones y se idealizaron diferentes cuestiones que podrían hacerse en un futuro como parte de dar mantenimiento al programa, algunos de estos planteamientos son:

a) Una de las cosas a mejorar a corto plazo es la parte de implementación de una interfaz gráfica o de algún medio de visualización más agradable para el usuario, esta interfaz ayudaría para hacer del programa más amigable con el usuario.

b) Otra cuestión a remarcar es el agregado de diferentes formatos de documentos de texto puesto que por el momento se limita al formato .txt además de adicionar otros métodos de encriptación, la inclusión de una mayor selección haría del programa más completo.

c) El tercer posible mantenimiento que podría requerir el programa sería la parte de verificar actualizaciones, ya sean por parte del lenguaje de programación, de los diferentes formatos de texto/criptografía o incluso del mismo sistema operativo pues en el caso de los formatos podría suponer un problema mayor para el programa pues supondría un cambio total en el funcionamiento interno del programa, lo que haría de nuestro programa obsoleto y es por eso que sería un importante punto a considerar

d) El cuarto es la actualización del código para atender complicaciones que puedan surgir por parte de los usuarios pues suele suceder que cuando se crea un programa, no se tiene en consideración algunos aspectos que pueden hacer de errores para algunos usuarios, algunos de estos errores pueden ser bugs, situaciones no planeadas por parte del programa, errores de compatibilidad con algún equipo por parte de algún usuario, entre otros.

e) El quinto y último punto a mencionar pero no menos importante es el mejoramiento y optimización de código de manera general pues al ser un proyecto creado a partir de programadores con poca experiencia y en formación da lugar a que el programa sea poco eficiente en algunos aspectos y que se podrían mejorar con la implementación de algunas metodologías.

   Una vez dicho lo anterior, sobre la cuestión de cuánto se esperaría cobrar por la realización de este programa y por futuros mantenimientos, una situación a plantearse consiste en la labor de un programador, que presentan varios retos, siendo principalmente el planteamiento y desarrollo en la solución de un problema un aspecto clave a considerar al momento de estimar algún honorario además de quien solicita el servicio del programador, horario de trabajo, fecha de entrega, factores externos como lo es consultar el servicio de demás lugares, el tipo de problema a resolver; esto formula la "pregunta definitiva"de responder pero diremos que, en este caso, se cobraría por el programa un monto de alrededor de 1500 a 3500 pesos aproximadamente pues se trata de un proyecto menor y que no requiere de una gran planificación de por medio.

6. Cambios realizados recientemente

Se realizaron los siguientes cambios:

a) Se hicieron cambios en la organización de las clases y subclases que componen el programa, esto con el fin de contribuir al mantenimiento e implementación de código.

b) Se optó por agregar un menú de opciones al programa con el fin de facilitar la visualización de los diferentes resultados.

c) Se reestructuró la organización de los archivos contenidos en Shamir-s-Secret-Sharing-Scheme/, se creó una carpeta de nombre "Pruebas"que, como su propio nombre lo indica, sirvió para la experimentación del programa, además se movieron los archivos prueba con terminación .txt a esta carpeta.

d) Se hizo un pequeño cambio de nombre al archivo .ENDEncryptApp.py", se renombró a .ENDEncryptApp.pyw", esto con el fin de hacer del programa un ejecutable

e) Se agregó la documentación al código del programa.

f) Se agregó el archivo "Proyecto_Aguirre_Leonardo_Valencia_Jonathan.ª "Shamir-s-Secret-Sharing-Scheme/", además de agregar contenido a README.md", en él se muestra cómo ejecutar el programa además de información adicional de cada archivo ubicado en el proyecto.

Puedes ver los cambios más a detalle en el apartado Commits en el GIT del proyecto:

https://github.com/WraithLion/Shamir-s-Secret-Sharing-Scheme.git