ENTREGABLE 1 - CRIPTOGRAFÍA Y SEGURIDAD INFORMÁTICA

Grupo 82

Iciar García Izquierdo - 100495789 - 100495789@alumnos.uc3m.es
Isabel Hernanz García - 100495778 - 100495778@alumnos.uc3m.es
Salvador Ayala Iglesias - 100495832 - 100495832@alumnos.uc3m.es

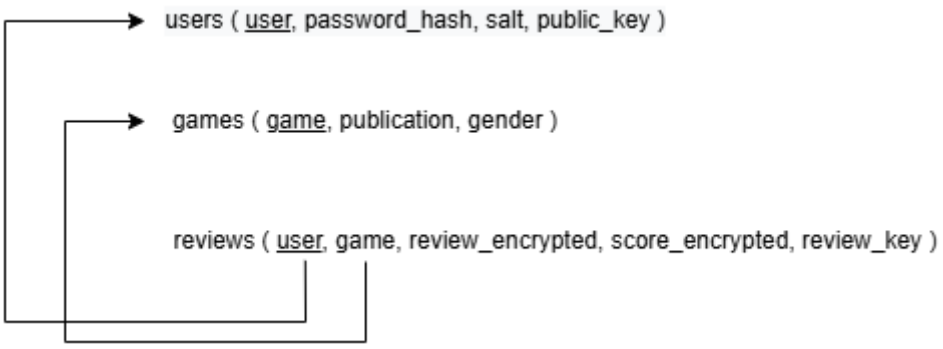
Índice:

1.- ¿Cuál es el propósito de su aplicación? ¿Cuál es su estructura interna?

La aplicación desarrollada consiste en un gestor de reseñas de videojuegos. El usuario podrá iniciar sesión o registrarse para poder escribir su reseña. Las reseñas contendrán un texto donde el usuario podrá escribir su opinión acerca del juego, y una puntuación que el usuario decidirá. La aplicación permite guardar reseñas para uno mismo, o enviarlas a otros usuarios.

Se cifrarán ambos el texto y la puntuación de cada reseña con una clave simétrica, y se introducirán en una tabla de una base de datos creada con "sqlite3" llamada "reviews". Junto con esas dos columnas, habrá tres columnas más: El usuario al que va dirigida la reseña, el juego del que trata; y la clave simétrica con la que se han cifrado el texto y la puntuación. Como queremos que no todo el mundo acceda a esas reviews, esta clave será cifrada con un algoritmo de encriptado asimétrico utilizando la clave pública del usuario destinatario. De esta forma, sólo él podrá descifrar la clave simétrica con su clave privada para poder acceder a su reseña.

La base de datos cuenta con la siguiente estructura interna:



La aplicación cuenta con un directorio dedicado únicamente a la creación de las ***clases*** (Usuario, Review y Juego). Serán útiles para organizar cada objeto que se use en adelante. Cuenta también con un directorio (*graphics*) dedicado a la **interfaz gráfica** de usuario, creada con "tkinter".

El resto de archivos están en el root principal del repositorio, de los cuales pueden destacar:

- *criptografia.py* es un archivo dedicado únicamente a la definición de funciones criptográficas que se encargan de todo lo relacionado con cifrar, descifrar y derivar la contraseña del usuario, así como almacenar dichas claves.
- *db_create.py* se encarga de la inicialización de la base de datos. Se debe ejecutar antes de iniciar la aplicación por primera vez.
- *main.py* es el script que se debe ejecutar para iniciar la aplicación.
- *gestionReviews.py* es un archivo con muchos métodos útiles para la gestión de las reseñas una vez el usuario las manda, o quiera acceder a ellas.
- *funciones_interfaz.py* controla la lógica de la aplicación, uniendo los elementos gráficos haciendo uso de diversas funciones.

2.- ¿Cómo se realiza la autenticación de usuarios? ¿Qué algoritmos ha utilizado y por qué? Detalle cómo se gestionan las contraseñas de los usuarios y si se generan claves a partir de éstas.

Al registrarse un nuevo usuario, se hacen varias comprobaciones: que no exista el usuario, que la contraseña tenga mínimo 8 caracteres y que la contraseña repetida coincida con la que el usuario ha introducido.

Una vez comprobado esto, se guarda el usuario como texto plano en la base de datos, junto con un hash (generado a partir de su contraseña y un salt generado aleatoriamente), el salt usado para dicho hash y su clave pública (generada a partir de su clave privada, generada aleatoriamente).

El hash de la contraseña se genera usando Scrypt, lo que nos permite evitar ataques de diccionario usando un salt aleatorio (generado en el momento) así como configurar diversas opciones para adaptar el algoritmo a las necesidades del desarrollador.

Las claves pública y privada se generan usando el algoritmo RSA, con un tamaño de clave de 2048, considerado seguro ante ataques de fuerza bruta. Una vez generadas la clave pública se guarda en la base de datos con el usuario. La clave privada se guarda en un archivo .pem, haciendo uso de la clave de autenticación de la aplicación para cifrar el contenido del archivo. Se usan dos funciones para escribir y leer de dicho archivo. Por supuesto, se crean claves nuevas por cada usuario.

3.- ¿Para qué utiliza el cifrado simétrico? ¿Qué algoritmos ha utilizado y por qué? ¿Cómo gestiona las claves? Explique los mismos aspectos si se utiliza cifrado asimétrico para este tipo de cifrado.

El cifrado simétrico se ha utilizado para cifrar tanto los textos de las reseñas como las puntuaciones. Siempre se cifra antes de actualizar la base de datos, y se ha elegido el algoritmo de cifrado AES (Advanced Encryption Standard). La razón de esta elección es la estandarización que tiene el algoritmo y su eficiencia, además de que permite elegir el tamaño de las claves usadas (esto ofrece flexibilidad en caso de que queramos cambiar ese valor sin tener que sustituir el algoritmo).

Sabemos que AES puede llegar a resultar menos eficiente que otros algoritmos simétricos como CHACHA20, pero como no tenemos pensado escalar la accesibilidad de esta aplicación a otros dispositivos con menos prestaciones como podrían ser los dispositivos móviles, este motivo no ha sido un factor limitante en nuestra decisión.

Además del cifrado simétrico, hemos añadido una capa más con la implementación de un cifrado asimétrico con RSA para que más de un usuario pueda acceder a una reseña concreta. Las claves secretas generadas en el algoritmo de AES son cifradas con la clave pública del usuario destinatario, por lo que sólo se podrá acceder a la clave simétrica si tenemos conocimiento de la clave privada que descifra estos datos.

Si hubiéramos elegido un sistema de cifrado asimétrico en sustitución de AES, tendríamos que haber tenido en cuenta dos claves para poder descifrar las reseñas (la privada y la pública), por lo que en la siguiente capa de encriptado (donde antes habíamos cifrado la clave simétrica) hubiéramos tenido que cifrar la clave privada del usuario para poder ser capaces de tener acceso a esa reseña. Creemos que, con la idea de implementación que esta aplicación tiene, esta opción hubiera sido más vulnerable que como se ha hecho. La combinación de cifrado simétrico para la reseña y asimétrico para el

intercambio de claves es bastante útil, y por tanto haber usado asimétrico para ambas cosas hubiera sido muy poco eficiente.

4.- ¿Para qué utiliza las funciones de códigos de autenticación de mensajes (MAC)? ¿Qué algoritmos ha utilizado y por qué? ¿Cómo gestiona la clave/s? Explicite si utiliza algoritmos de cifrado autenticado y las ventajas que esto ofrece

Las funciones de códigos de autenticación de mensajes han sido utilizadas por tres motivos principales: garantizan la integridad de los mensajes recibidos por el receptor, en caso de que los datos hayan sido modificados a lo largo de la transmisión, se detectará. Además, el uso MAC también proporciona una manera fiable de verificar que el contenido recibido proviene de una fuente legítima que tiene en posesión la clave privada necesaria para utilizarlo. Por último, es una gran medida contra ataques pues proporciona ayuda a la hora de proteger los datos pues notifica si el mensaje ha sido cambiado.

A la hora de elegir algoritmos nos decantamos por HMAC (Hash-based Message Authentication Code) debido a su gran aceptación y uso. Este algoritmo en concreto requería una función hash criptográfica, que en nuestro caso elegimos SHA-256, junto con una clave secreta.

A la hora de gestionar las claves necesarias para trabajar con HMAC, son generadas aleatoriamente para cada review y posteriormente guardadas en un archivo .pem codificadas en base 64, lo que garantiza su protección y facilita su uso posterior.

5.- Github

Enlace al repositorio de github: https://github.com/100495778/iciar-isa-salva-criptografia.git