Proyecto 1 – Comunicaciones Seguras

Competencias a desarrollar

- Implementar cifrados simetricos
- Implementar cifrados asimétricos
- Repasar los conecptos básicos de cifrados

Problemas a resolver

El proyecto tiene como objetivo diseñar e implementar un sistema de comunicaciones seguras donde los usuarios pueden registrarse, intercambiar mensajes cifrados y participar en chats individuales o grupales de manera segura.

Cada usuario deberá registrarse en el servidor, proporcionando su nombre de usuario y generando un par de llaves pública y privada. La llave pública se compartirá a través de una API para que otros usuarios la utilicen al enviar mensajes cifrados. La comunicación se realizará utilizando cifrado asimétrico para garantizar la confidencialidad de los mensajes.

Además, se permitirá la creación y gestión de grupos de chat, donde los mensajes grupales estarán protegidos mediante cifrado simétrico AES-128.

El proyecto incorporará conceptos de criptografía simétrica y asimétrica, autenticación segura, almacenamiento seguro de llaves, y una interfaz de usuario HTML para facilitar la interacción entre los usuarios y el sistema.

Este proyecto no solo brindará a los estudiantes una comprensión práctica de la seguridad en las comunicaciones, sino también la oportunidad de aplicar y desarrollar habilidades en programación segura y gestión de claves criptográficas.

La documentación y presentación en clase permitirán a los equipos compartir sus experiencias y soluciones implementadas.

Requerimientos del sistema:

Base de datos:

Con el objetivo de almacenar la información se utilizarán tablas de base de datos, para este proyecto pueden utilizar base de datos relacionales o no relacionales.

Usuario	
1	id
2	public_key
3	username
4	fecha_creación

Mensajes	
1	id
2	mensaje_cifrado
3	username_destino
4	username_origen

Grupos	
1	id
2	nombre
3	usuarios
4	contraseña
5	clave_simetrica

Mensajes_Grupos	
1	id_grupo
2	author
3	mensaje_cifrado

Usuarios:

- Tiene como objetivo registrar los nuevos usuarios que se regtistren al chat.
- Almacena un llave publica en base64.
- Almacena el username.
- Almacena la fecha que fue creada.
- La llave publica nos permitirá realizar las comunicaciones para los chats individuales.

Mensajes:

- Tiene como objetivo almacenar cada una de las conversaciones que se realizan entre usuarios.
- Almacena el mensaje cifrado con la llave pública del usuario
- Almacena el nombre del usuario destino
- Almacena el usuario de origen.

Grupos:

- Tiene como objetivo crear una conversación grupal entre un grupo de usuarios.
- Almacena un nombre que lo identifique
- Almacena un listado de usuarios que pertenecen al grupo.
- Almacena una contraseña cifrada en AES, que nos permitirá autorizar eliminar el grupo.
- Una clave simetrica en AES que servira de comunicación general para todos los integrandes del grupo para poder descifrar los mensajes cifrados.

Mensajes de Grupos:

- Tiene como objetivo almacenar cada uno de los mensajes que se generen dentro del grupo para esto se debe almacenar el author del mensaje y el mensaje cifrado.
- El mensaje cifrado es el que se descifrará utilizando la clave simétrica de la tabla Grupos.

API:

GET:

- Con el fin de poder acceder a la información que se estará almacenando en la base de datos se requiere los siguientes endpoints:

- /users/{user}/key:
  - Obtiene la llave pública del usuario en base 64
- /users
  - Obtiene el listado de usuarios que estan registrado en el servidor
- /messages/{user_ origen}/users/{user_destino}
  - Obtiene el listado de mensajes entre 2 usuarios
  - Para poder leer los mensajes del usuario es necesario tener la llave pública
- /groups
  - Obtiene el listado de grupos disponibles
- /messages/groups/{group}
  - Obtiene el listado de mensajes que pertenecen al grupo
  - Para poder ver los mensajes es necesario tener la llave simétrica

POST:

- /users
  - Almacena/registra un nuevo usaurio con su llave pública
- /messages/{user_ destino}
  - Guarda un mensaje para el usuario destino
- /groups
  - Guarda un nuevo grupo
- /messages/groups
  - Guarda un mensaje para el grupo destino

PUT:

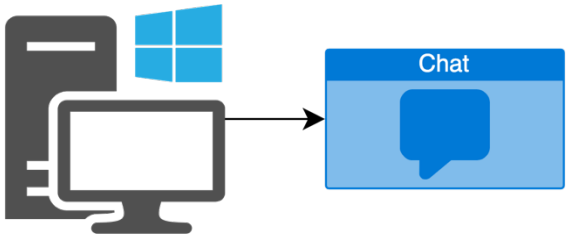
- /users/{user}/key:

- Actualiza la llave pública del usuario en base 64

DELETE:

- /users/{user}/key:
  - Elimina la llave del usuario dentro
- /users/{user}:
  - Elimina el usuario
- /groups/{group}
  - Elimina el grupo completo, para esto es necesario validar que la contraseña sea la correcta.

CHAT:



El aplicativo puede realizarse en cualquier tecnología que permita intercambhio de datos por medio de HTTPS/ Rest api.

Para esto necesitará desarrollar lo siguiente:

1. Iniciar sesión (registrarse generándo nuevas claves o utilizar una existente)
2. Visualidar Un listado de usuarios
3. Seleccionar un usuario
   a. Ver los mensajes enviados al usuario (Cifrarlos con llave pública)
   b. Ver los mensajes que fueron enviados a su usuario. (Desencriptar usando la llave privada)
4. Visuarlizar Un listado de grupos
   a. Crear un grupo
   b. Crear un chat grupal.

Para esto tome en cuenta que sera necesario tener la llave privada en un sitio seguro para poder descifrar los mensajes que reciba.

Así mismo por cada grupo deberá crearle una contraseña para poder luego eliminarlo, tambíen puede crear una llave aleatoria que sera utilizada para el AES 128 con el fin de descifrar y cifrar todos los mensajes del grupo.

Entregables:

- Documento de diseño
    - Una descripción detallada de la arquitectura del sistema, incluyendo cv≥mo se manejan las llaves, cómo se almacenan, y los mecanismos de cifrado y descifrado implementados.
- Código fuente en Github
- Interfaz HTML/CSS/JS
- Informe de Desarrollo
    - Un informe que describa los desafíos encontrados durante el desarrollo, las soluciones implementadas y cualquier mejora futura que podrían considerar.
- Pruebas y Validación
    - Un conjunto de casos de prueba que demuestren la funcionalidad y seguridad del sistema. Esto podría incluir casos de prueba para el cifrado, descifrado, manejo de llaves y otras características críticas.
- Publicación de proyecto
    - Su proyecto publicado en alguna tecnología en la núbe o empaquetado en un docker