SecureBox - Documento ERS

1. Introducción

SecureBox es un sistema de almacenamiento seguro de secretos diseñado para garantizar la confidencialidad, integridad y disponibilidad de los datos. El sistema permite gestionar múltiples contenedores de secretos, protegiéndolos mediante cifrado AES y autenticación robusta. Además, ofrece funcionalidad de respaldo en la nube (Google Drive) para recuperación ante pérdidas.

2. Requisitos Funcionales

2.1. Gestión de Autenticación

[RF-1] El sistema permite la autenticación de usuarios mediante contraseña, utilizando derivación de claves con la función scrypt y un salt generado de forma segura mediante os.urandom.

[RF-2] El sistema proporciona un mecanismo de retraso exponencial ante intentos fallidos repetidos.

[RF-3] El sistema almacena de forma segura la clave derivada y el salt en un archivo protegido con permisos restringidos.

2.2. Gestión del Vault

[RF-4] El sistema permite la creación, edición y eliminación de contenedores.

[RF-5] El vault se cifra usando AES-256 en modo CBC con una DEK (Data Encryption Key) que se obtiene a partir de una KEK (Key Encryption Key) derivada de la contraseña del usuario.

[RF-6] El sistema verifica la integridad mediante HMAC-SHA256 en los procedimientos de carga y descarga del vault.

[RF-7] El sistema permite la persistencia del vault en un archivo local que permanece cifrado en disco en todo momento.

2.3. Integración con la Nube

[RF-4] El sistema permite la autenticación OAuth2 con Google Drive y gestión de tokens de acceso.
[RF-5] El sistema permite la subida del vault cifrado a Google Drive en una carpeta dedicada.
[RF-6] El sistema permite la descarga de un vault desde Google Drive para recuperación de datos.

2.5. Interfaz de Usuario

[RF-7] CLI interactiva con menús jerárquicos (menú principal, vault, nube).
[RF-8] Visualización de contenedores con formato legible (ID, nombre, secretos).

3. Requisitos No Funcionales

3.1. Seguridad

[RNF-1] Resistencia a ataques offline mediante uso de scrypt (parámetros: N=2^14, r=8, p=1).
[RNF-2] Cifrado doble para los secretos: DEK (clave de datos) protegida por KEK (clave derivada de la contraseña).
[RNF-3] La autenticidad del vault se verifica mediante HMAC en cada operación de carga y descarga.
[RNF-4] Archivos locales (vault.json, auth.json) con permisos restringidos (0o600).

3.2. Código

[RNF-5] Diseño orientado a objetos con separación de responsabilidades.
[RNF-6] Documentación interna en cada función y clase.

3.3. Disponibilidad

[RNF-7]: Copias de seguridad manuales en Google Drive para recuperación ante pérdidas locales.