Projet-C Cloud et Partage de fichiers

Fonctionnalités

Nom : The Untitled Cloud Storage - TUCS
Date de rendu : 07 février 2021

Projet Open-Source
Langue : Anglais (prévoir une possibilité d'ajouter d'autres langues facilement avec un fichier de config)

A noter que les listes des bibliothèques et fonctionnalités sont amenées à changer et à se développer.

Bibliothèques :
https://github.com/openssl/openssl
https://github.com/warmcat/libwebsockets
https://github.com/mongodb/mongo-c-driver
https://github.com/mirror/ncurses
SGBD - MongoDB : https://www.mongodb.com/fr

Gestion d'utilisateurs
Les utilisateurs doivent s'inscrire sur la plateforme : nom d'utilisateur et mot de passe.
(ce qui implique que si l'utilisateur perd son mot de passe, c'est perdu, pas de récupération). Des clefs privées seront générées et associées à l'utilisateur.

Envoi de fichiers
Les utilisateurs inscrits sur le service pourront envoyer des fichiers depuis leur machine à partir d'un client de leur choix.

Téléchargement de fichiers
Les fichiers envoyés par des utilisateurs seront téléchargeables à tout moment.
La seule façon d'accéder au fichier est d'avoir au préalable un compte sur la plateforme, puis que l'utilisateur qui possède le fichier aie créé un URI (Unified Resource Identifier).

Lorsqu'un utilisateur se connecte pour la première fois à la plateforme sur une nouvelle machine, il lui sera proposé de synchroniser tous ses fichiers automatiquement ou manuellement.

Utilisation de web sockets pour transférer les fichiers entre le serveur et le client.

Stockage côté serveur sous la forme :
/<user>/<folder uid>/.../<file uid>
Les noms des fichiers sont chiffrés par le client avant d'être envoyés au serveur, seul le client peut déchiffrer avec sa clé privée.

Partage de fichiers
Une fois les fichiers envoyés sur le réseau, ces derniers pourront être rendus accessibles à d'autres utilisateurs.

- Fonctionnement du partage de fichier :

A l'origine, le fichier est chiffré avec une clef utilisateur immuable. Quand on met en place le partage d'un fichier, ou d'un dossier, on a un processus qui effectue de nouveau un chiffrement du fichier avec une clé partageable. Pour des raisons de sécurité, tout se passe du côté client.

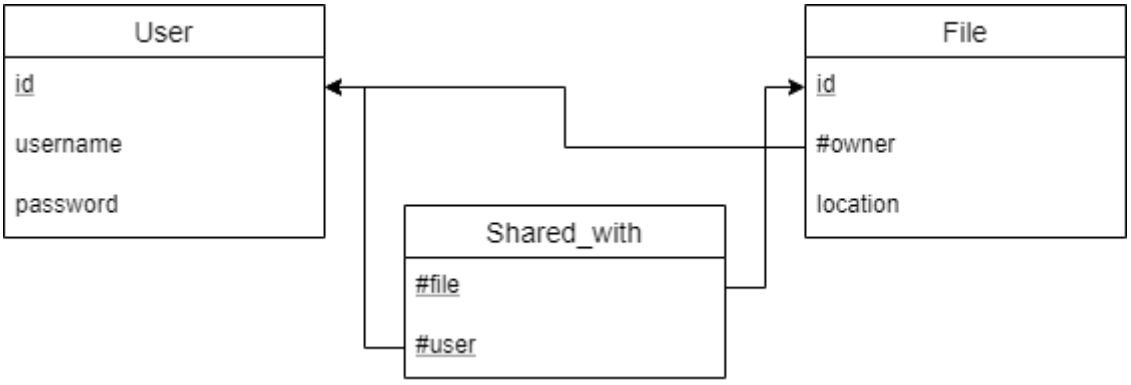
L'utilisateur pourra choisir parmi les différentes clés qu'il aura créé pour le partage du fichier.

Chiffrement des données
Pour garantir une sécurité complète, toutes les données privées seront chiffrées.
Une clé privée par fichier à partager avec le fichier lors du partage.
Un utilisateur peut créer plusieurs clefs privées.
Chiffrement des données avec la bibliothèque openssl, si possible avec accélération GPU.

Accès client
Accès via client (développé en C) interface graphique développée avec l'API ncurses et utilisant la librairie libwebsocket.
Les utilisateurs auront la possibilité de développer des clients custom : mise à disposition d'une documentation légère sur la mise en place des sockets côté serveur.
Possibilité d'exporter le fichier de config qui contient les clé de chiffrement.

Serveur de stockage
Le serveur de stockage utilise une base de données pour stocker les informations sur les utilisateurs (clé de connexion, fichiers envoyés, ...). Le client se connecte au serveur pour envoyer et télécharger ses données.

Schéma de base de données



Nous souhaitons utiliser MongoDB comme SGBD.
A noter que MongoDB n'a pas besoin de schéma, les données seront donc organisées en "documents" avec 1 document par utilisateur inscrit contenant toutes les informations de ce dernier.