Rapport

Projet de Réseaux et Systèmes
Find the Cat

Wenjia TANG
Cécile de LAPASSE

18 décembre 2022

Sommaire

1 Introduction

L'objectif de ce projet consistait à écrire un programme qui permette de retrouver un fichier dans une arborescence. Sept options de base devaient être implémentées, comme retrouver le fichier selon son nom, sa taille ou son extension. Des extensions pouvaient aussi être réalisées parmi une liste donnée.

J'aimerais préciser (Cécile) que sur le dépôt Git, on peut voir des commits venant de Steven TRINH. Ma VM ne démarrant plus, j'ai pu continué d'implémenter les dernières finitions sur son ordinateur, et même en ayant renseigner mes identifiants en faisant le push, ils apparaissent comme étant de lui. Ce sont bien mes commits et mon travail qui a été rendu à son nom malencontreusement car je ne pouvais pas effectuer mon travail sur mon propre ordinateur.

2 Conception

Dans le projet, nous avions deux points essentiels à implémenter : le parcours de l'arborescence et le traitement de la ligne de commande.

Pour le parcours de l'arborescence, nous avons utilisé la structure de données préexistante *dirent* qui nous permet justement de parcourir les dossiers de l'arborescence et d'obtenir les informations pertinentes aux options, par exemple le nom du fichier ou du dossier avec l'attribut *d_name* entre autres. Nous avons aussi utilisé la structure *stat* pour connaître par exemple la taille des fichiers avec la méthode *st_size*.

Une fois les flags donnés en ligne de commande reconnu (ou non) après les avoir comparés avec les différents flags possibles, on doit effectuer une sélection sur les fichiers par rapport aux options demandées. Pour sauvegarder cette liste d'options, nous avons finalement choisi d'utiliser une structure de données classique, une liste simplement chaînée, où chaque élément sauvegarde un indice (int, celui de la position de l'option dans un tableau de notre implémentation : il sert à exécuter la fonction correspondante), le nom de l'option (String) et son paramètre (String). En parsant la ligne de commande, on récupère donc le nom de l'option et un paramètre qu'on sauvegarde dans cette structure. Dans le cas de l'option *ftc*, où l'on peut donner plusieurs mots en paramètre, tous ces mots sont mis dans une seule String que l'on passe en paramètre de l'option.

De même, pour la liste finale des fichiers (ou dossiers) qui répondent à toutes les options, nous avons créé une autre structure de liste simplement chaînée où chaque élément contient le chemin du fichier (ou dossier).
L'option *mime* a été implémentée en utilisant la librairie *MegaMimes*.

3 Difficultés rencontrées

Les principales difficultés rencontrées dans l'implémentation de notre projet concernent le parcours de l'arborescence et l'implémentation de l'option *-mime*.

3.1 Parcours de l'arborescence

Le parcours de l'arborescence est fondamental pour la réalisation du projet *Find the Cat*. Cependant, nous étions bloquées par ce parcours notamment à cause du manque de connaissances de la structure C : *dirent.c*. Lorsque nous lancions notre programme, le nombre de dossiers et fichiers sous un répertoire était toujours plus grand que le nombre obtenu en faisant "ls" sur un répertoire. En effectuant plusieurs tests, nous avons réussi à localiser le problème de notre programme : nous avions également compté les fichiers ou dossiers cachés. Cela est dû à l'appel de la fonction *readdir* sans ajout de contraintes. En effet, la fonction *readdir* en C peut consulter tous les fichiers et dossiers même s'ils sont cachés.

Pour résoudre ce problème, nous avons donc ajouté des contraintes. Après que la fonction *readdir* ait consulté un fichier, nous récupérions le nom du fichier ou du dossier. Si ce nom commence par un ".", nous n'allons pas afficher son chemin. Pour le dossier courant (".") et le dossier précédent (".."), nous avons effectué la même opération.

D'autre part, en implémentant les différentes options les unes à la suite des autres, ce parcours de l'arborescence devait être modifié, par exemple pour traiter les dossiers quand on a l'option *dir* ou quand on veut sélectionner les fichiers qui répondent à toutes les options demandées. Ces ajouts ont demandé la restructuration et l'amélioration du parcours de l'arborescence, ce qui créait toujours différents problèmes ou en mettait en lumière. C'était une des parties particulièrement chronophages.

3.2 Implémentation de *-mime*

Le deuxième problème rencontré était l'implémentation de l'option *-mime*. En effet, la difficulté était de trouver une librairie utilisable et complète. Nous avions trouvé une base de données qui s'appelle *magic.h*. Cependant, c'est une base de données qui détecte le type selon l'encodage du fichier, ce qui ne convient pas au sujet.

Pour résoudre ce problème, comme nos recherches n'avançaient pas, nous avons demandé à un autre groupe qui avait réussit à trouver une librairie utilisable. Cette dernière s'appelle *Megamimes*.

4 Extensions réalisées

Pour les extensions, nous avons réalisé l'option *-perm* qui permet de chercher les fichiers selon leurs permissions.

4.1 Option *-perm*

Nous avons utilisé l'élément *st_mode* de la structure *stat*. Cet élément nous permet de connaître les permissions d'un fichier. En utilisant *st_mode&0777*, nous pouvons obtenir les informations des permissions en décimal. Cependant, les permissions recherchées sont passées en format octal. Par conséquent, nous avons décidé de tout transformer en décimal et de les comparer. Si les permissions données en paramètres sont égales à celles données par *st_mode&0777*, alors la fonction va retourner *true*, sinon, elle retourne *false*. Cela nous a ainsi permis d'isoler les fichiers qui satisfont les conditions passées en paramètre et de les afficher sur le terminal.

5 Nombre d'heures passées sur le projet

	Wenjia	Cécile
Conception	8 h	2h
Implémentation	18 h	27h
Test	2 h	2h
Rédaction du rapport	3 h	2h30