Rapport RS 2017

HENRY Thibault
OHANIAN Raphaël

CONTENTS

INTRODUCTION

Dans le cadre du module Réseau - Système (RS) de 2ème année, nous avons été amenés à réalisé un projet nommé "tesh". Le but de ce projet est de créer un tesh fonctionnel. Celui-ci doit ce comporter comme un shell linux classique pour les fonctions demandée dans le sujet.

L'objectif de ce projet est également de nous faire revoir et mettre en pratique le cours de système, notamment la manipulation de processus.

Ce projet est à effectuer par groupe de deux. Il faut donc apprendre à travailler en binôme.

Ce projet sera évalué à l'aide de test (donc pas un humain). Pour nous aider, plusieurs tests blancs serons réalisés.

CONCEPTION

Notre tesh est un exécutable. Il sera lancé un partir d'un shell linux et permettra de gérer différentes commandes.

De plus, certaines commande peuvent s'écrire de différentes façon en shell. Par exemple, cmd1 || cmd2 équivault en shell Unix à cmd1|| cmd2, cmd1 ||cmd2 ou encore cmd1||cmd2. Cependant, nous ne traiterons dans notre tesh que le cas cmd1 || cmd2, et ce, pour tous les caractères de redirections.

Il nous faudra pouvoir récupérer le nom de l'utilisateur courant ainsi que l'hostname de la machine afin de pouvoir afficher un prompt. Ces données peuvent etre récupéré grâce à des commandes déjà présentes en C.

Le tesh devra être en mesure de s'exécuter en mode interactif et non-interactif.

En mode interactif, l'utilisateur écrit lui même les fonctions. De plus, le shell affiche le prompt avant chaque commande.

En mode non-interactif, si un fichier contenant des commandes à exécuter est passé en paramètre, le tesh doit pouvoir les exécuter. De plus, en mode non-interactif, le shell n'affiche pas de prompt.

Le tesh doit pouvoir réagir à l'option -e (sortie sur erreur). Si cette option est passé à tesh, il doit s'arrêter dès qu'une commande termine avec un code de retour différent de 0.

Enfin, le doit doit réagir à l'option -r (readline). Si cette option est passé à tesh, alors le tesh doit utilisée la bibliothèque readline pour permettre l'édition interactive de la ligne de commande, et la gestion de l'historique.

DIFFICULTÉS RENCONTRÉES

La première difficulté rencontré a été d'assimilé le sujet, de comprendre les attentes et d'essayer d'anticiper les difficultés.

Une fois le sujet bien pris en main, il a fallut comprendre les structures. Il n'a pas été facile de comprendre le fonctionnement d'un shell.

Une remise à niveau sur les pointeurs a été nécessaire. En effet, nous avons manipulé beaucoup de chaînes de caractère, interprété comme des pointeurs en C.

La création du parser nous a posé problème. Cette fonction a d'ailleurs été re-codé plusieurs fois. Cependant, une modification de celle-ci entraînait souvent une modification de tout le reste du code pour s'adapter au type de sortie du parser. Nous avons perdu beaucoup de temps à cause de celui-ci.

Le travail en équipe nécessite un temps d'adaptation. En effet, chaque personne à un type de rédaction qui lui est propre. Il faut donc un certain temps pour s'adapter à son équipe et produire du code que toute l'équipe pourra comprendre sans trop de difficultés

TEMPS PASSÉ SUR LE PROJET

Le tableau ci-dessous correspond aux heures de travail que nous avons passé sur le projet. En plus de l'étape de programmation, nous avons consacré du temps pour essayer de reproduire les tests localement pour optimiser notre rendu.

Tâches	Thibault	Raphaël
Conception	12h	15h
Implémentation	20h	15h
Tests	2h	2h
Rapport	0h	2h
Total	34h	33h

BIBLIOGRAPHIE

Voici les sites nous ayant aidé à comprendre le sujet :

https://openclassrooms.com/forum/sujet/mini-shell-en-c-environnement-66072

http://chrtophe.developpez.com/tutoriels/minisysteme/

https://members.loria.fr/LNussbaum/rs.html