

Projet de réseau
L3 Informatique
2021-2022

Projet
Serveur HTTP

Dombry Baptiste
Dourlen Maxime
Groupe TP3

Sommaire :

- I. Introduction
- II. Choix d'implantation
- III. Limites et problèmes rencontrés
- IV. Conclusion

I. Introduction

Lors du cours de réseau, il nous à été demandé de faire ce projet. Ce projet a pour but d'écrire serveur HTTP, ce dernier doit pouvoir accepter des connexions entrantes TCP, recevoir des requêtes d'un navigateur et doit pouvoir répondre à la requête reçue. Les requêtes reçues doivent correspondre aux normes, avec une ligne de commande, suivi d'en-tête de requête et une ligne vide. Seul la ligne de commande est obligatoire pour une requête correcte. Le serveur doit analyser la requête du client et envoyer une réponse en fonction de la requête reçu. Si la demande du client peut-être réalisé alors la réponse du serveur doit contenir une ligne de statut, suivi d'un entête de réponse, une ligne vide et le contenu demandé par le client. Dans le cas contraire, la ligne de statut contiendra le code de l'erreur rencontré.

II. Choix d'implantation

Afin de réaliser ce serveur nous avons utilisé les bibliothèques `adresse_internet` et `SocketTCP` développé lors des séances de TP. Ces bibliothèques servent respectivement à obtenir les informations nécessaires à une connexion, les stocker dans une structure permettant d'y accéder facilement, et créer un socket, l'utilisent, l'enregistrer dans une structure adaptée.

La bibliothèque `adresse_internet` nous permet facilement de créer une structure `sockaddr` avec le port fourni, nous avons utilisé cette implantation car elle nous permet de faciliter la gestion des différentes structure et type utilisé dans le serveur.

La bibliothèque `SocketTCP` utilise la bibliothèque précédente `adresse_internet`, `SocketTCP` nous permet de créer, utiliser, et stocker les sockets. Nous avons utilisé cette implantation parce qu'elle nous permet la simplification des appels système ainsi que les données utilisées pour ces derniers.

Dans le développement du serveur, nous avons fait le choix de faire dans le main tous les appels aux fonctions de création de socket et de mise en écoute puis une fois une connexion reçue, lancé un thread dédié à toutes les tâches en rapport avec la socket d'écoute. Le thread est créé de manière détaché parce qu'il nous est pas nécessaire de vérifier le retour du thread, en cas d'erreur la connexion se ferme et rien n'est envoyé.

Dans le développement du thread, nous avons choisi de découper l'analyse de la requête en plusieurs parties, tout d'abord on récupère les informations principales et nécessaires à la réponse, ensuite ont vérifié si la requête reçue ne contient pas d'erreur. Si elle contient une erreur alors on renvoie le code d'erreur qui est détecté dans la fonction erreur.

III. Limites et problèmes rencontrés

Lors du développement, nous avons rencontré plusieurs problèmes :

- Des fuites de mémoire ce qui menait à un arrêt du programme, nous avons pu résoudre le problème grâce à valgrind.

- Des zones mémoire allouées à chaque requête qui ne se libéraient pas avec un free(), nous avons réglé le problème en déplaçant l'allocation en dehors de la fonction.

- La détection et création du type MIME, nous avons donc créé une fonction qui avec l'extension indique le type et le sous-type MIME.

- Lors de la connexion d'un client le serveur lisait directement sur le socket mais si le client n'envoyait aucune requête le thread en charge de ce client restait bloqué en attendant la requête, ce qui pouvait causer un déni de service. Nous avons ajouté un système de timeout qui au bout de 3 secondes ferme la connexion ainsi que le thread.

Nous avons également trouvé des limites à notre projet, ses limites sont dans le cas où beaucoup de clients demandent des pages lourdes en même temps le temps de réponse va être considérablement allongé.

IV. Conclusion

Pour conclure, nous avons trouvé cet enseignement et ce projet très enrichissant dans notre cursus de licence Informatique. Nous avons donc appris la manipulation des sockets internet en mode non connecté et en mode connecté en mode TCP. Ce projet nous ouvre les portes sur beaucoup d'opportunité de projet personnel comme un serveur HTTP local.