Programmation Réseau Système

Implémentation d’un serveur TCP

Pierre Ripoll
Jean-Nicolas Winter

Première implémentation

2 processus :

- un père qui va **gérer les connections** sur le port “public”
- les fils qui **gèrent le transfert de fichiers** pour son client dédié, puis une fois fini arrête son processus.

1 problématique :

gestion lors de la connection car il faut **préparer le fils** avant la transmission au client du port sur lequel il pourra se connecter

Résolution :

Coordination entre le père et le fils grâce à **des signaux**, et uniquement lorsque le fils est **prêt à écouter** sur le **numéro de port dédié**, le père envoie ce numéro au client

Gestion du transfert de fichier

Pour le transfert de fichier, on a décidé d'optimiser le programme pour l'envoie et la réception des paquets :

Utilisation de Threads (pour pouvoir faire des actions en parallèle).

Utilisation d'un buffer circulaire qui contient : **numéro du paquet**, **paquet à envoyer**, **durée de validité du paquet** (timeout).

Pour chaque case du buffer circulaire, on a un **Thread** qui **décrémente le "Time"** jusque zéro

2 Threads :

-**send** qui vérifie chaque case du buffer circulaire entre deux pointeurs "**start**" et "**stop**" et envoie ceux dont le "Time" est zero

-**receive** qui écoute le client et met "ack" selon les **acknowledgement reçus** dans le buffer circulaire pour le paquet correspondant et déplace le **pointeur "start"** (indique au *send* qu'il n'a plus besoin d'envoyer ceux reçu)

Main : gère le buffer circulaire = **enlève les paquets** dont on a reçu les "ACK" et **met les nouveaux paquets** à envoyer dans le buffer, puis déplace le **pointeur "stop"** (indique que le *send* peut envoyer d'autres paquets)

Recherche d’optimisation

fread : plus rapide ou pas ?

on chargeait le **fichier en entier** dans un buffer : problème si le fichier était trop grand

maintenant on charge petit à petit (en espérant que fread soit performant)

résultats : *fread* moins performant, plus rapide de charger le fichier en entier (débit de 950 != 800)

➜ Implémentation : si le fichier est **trop grand** (**>100 Mo**) on charge **petit à petit** le fichier, sinon on le charge en entier.

Plusieur paramètres qui influence le débit

Taille du segment à envoyer : on a une **MTU de 1500,** et le client peut accepter au maximum des segment de 1500, on envoie en tout un **segment de taille maximum 1500**

Taille Buffer Circulaire :

Taille Fenêtre d'Envoi :

Temps avant retransmission : On peut mettre le **RTT** (temps aller-retour). On pourra l'**évaluer** en fonction du premier paquet transmis.

Fast Retransmit : Transmet le paquet suivant directement après "*x*" ACK reçu de suite

Faire les tableaux de performances

Implémentation

Slow Start :

Objectif : avoir rapidement la taille de la fenêtre d’envoi optimale

Congestion avoidance :

Éviter de saturer le réseau, modifie la taille de la fenêtre d’envoi (la réduit)

Fast Retransmit :

Permet de renvoyer directement le prochain paquet si l’on reçoit plusieurs fois d’affilé un même ACK

Spécificité :

client1

client2 ➜ énormément d’ACK du même paquet, utilité du Fast-Retransmit

multiclient