Linux : Appels Systèmes - I2181 (8)

8.2 Exercice sur les appels systèmes : `poll`

Réflexion après avoir fait l'exercice :

- Que fait le processus serveur pendant son temps libre (c'est-à-dire lorsqu'il ne réceptionne pas de message des clients) ?

Pour l'instant le serveur ne fait rien de son temps libre. L'utilisation de *poll* permet simplement de traiter les réponses des clients dans leur ordre d'arrivée (Ex: réponse joueur2, réponse joueur1) et non de manière séquentielle (réponse joueur1, réponse joueur2, …).

- Peut-on mieux exploiter « le temps libre » du processus serveur ?

Le serveur pourrait profiter de son temps libre pour surveiller son socket d'écoute. Il peut alors soit:
- Poliment répondre aux clients qui essaient de se connecter de revenir plus tard car une partie est en cours
- Préparer déjà la prochaine partie c'est-à-dire déjà accepter des inscriptions pour une prochaine partie

- Augmenter le timeout du poll est-il intéressant ?

Le timeout est en millisecondes.

Avec un timeout de 0 :
poll n'attend pas et jette un simple coup d'œil aux files descriptors qu'il doit surveiller. Cette solution permet un traitement rapide dès qu'un client envoie qqch MAIS envoie énormément d'appels systèmes poll au système d'exploitation.

Avec un timeout de 60000 :
Poll va surveiller les files descriptors pendant maximum 60 secondes. Si aucun client ne se présente pendant 60 secondes, le serveur restera bloqué pendant 60 secondes. Donc ici au pire nous allons rester bloqués pendant 60 secondes que le serveur puisse faire qqch d'autre que s'occuper des clients (écrire un log, …) MAIS nous n'utiliserons qu'une seule fois l'appel système poll

En conclusion, il est intéressant d'avoir un timeout ni trop petit, ni trop grand afin d'obtenir un bon compromis (pas trop d'attente et pas trop d'appel à poll).
Un timeout de 15 secondes par exemple.