Introduction

Auteurs : Tiffany Bonzon et Jérôme Arn

Description des fonctionnalités du logiciel

Le logiciel remis est une modification du logiciel donné afin de pouvoir travailler sur plusieurs threads. Il a donc les mêmes fonctionnalités qu'initialement avec, cette fois-ci, la possibilité de craquer des hash MD5 sur plusieurs threads.

Approche du problème

Les tâches de gestion de la barre d'avancement et de répartition des calculs entre les threads ont été attribuées au thread principal. Tandis que la génération de tous les hashs sur un intervalle donné, la comparaison avec le hash a cassé ainsi que l'incrémentation du compteur principal ont été attribués aux threads. La quantité de code relativement importante qui compose les threads s'explique car elle est intrinsèquement liée entre elle. Néanmoins la partie critique modifiée et lue par le thread principal et les threads de génération de hashs est restreinte à une variable.

Cette variable est globale à tous les threads qui génèrent des hashs et elle est protégée par un mutex lors de son incrémentation et lors de sa lecture par le getter.

Dans un premier temps, nous avions pensé facilité le passage des nombreux arguments en transmettant un objet Threadmanager avec des getter et des setter qui seraient ensuite utilisés dans les threads. Mais après différent tests, nous avons pu remarqué que cela ralentissait considérablement les calculs. Pour cela nous avons transmis tous les arguments en copie.

Test du programme

En règle générale, tous nos tests ont menés à un résultat concluant. Mais de manière aléatoire et sans que nous ayons pu en déterminer la raison, deux tests ont finis de manière inattendue. Nous n'avons pas pu reproduire la situation en mode debugg.

Tests

Mot de passe de 3 caractères (***)

On peut voir dans ces exemples que plus on augmente le nombre de threads, plus le temps diminue.

- 1 Thread 570 ms
- 3 Threads 360 ms
- 7 Threads 280 ms

Mot de passe de 4 caractères ()

On peut voir dans ces exemples que plus on augmente le nombre de threads, plus le temps diminue.

- 1 Thread 31947 ms
- 3 Threads 12087 ms
- 7 Threads 5906 ms

Mot de passe de 5 caractères (*****)

- 1 Thread environ 28 minutes
- 3 Threads environ 14 minutes
- 7 Threads environ 10 minutes

Nous avons également testé des mots de passes n'étant pas aux extrémités
de l'espace de mots de passes.

Mot de passe de 3 caractères (Mix)

- 1 Thread 281 ms
- 3 Threads 111 ms
- 7 Threads 300 ms

Mot de passe de 4 caractères (test)

- 1 Thread 9434 ms
- 3 Threads 10823 ms
- 7 Threads 8805 ms

Mot de passe 5 caractère (swIss)

- 1 Thread environ 10 minutes
- 3 Threads environ 12 minutes
- 7 Threads environ 9 minutes

Nous avons également testé le cas ou le mot de passe n'est pas trouvé (par exemple en passant le hash du mot de passe "test", puis en demandant au programme de chercher un mot de passe à 1 caractère) pour s'assurer que la bar de progrès fonctionne bien comme voulu. Le fait d'avoir un sel ne change rien au temps ou à la complexité, les cas testés ne sont donc pas discutés.

Conclusion

On peut constaté que dans la majorité des cas, si on a plus de threads le temps de cassage est plus rapide. Mais dans d'autre cela va plus lentement. Cela peut venir de l'ordonnancement des threads et la position du mot recherché dans l'intervalle que doit calculé le thread.