Mutualisation entre machines virtuelles de la mémoire inutilisée.

Encadrants : Julien Sopena, Sébastien Monnet et Maxime Lorrillere
Contact : prénom.nom@lip6.fr

PSAR 2014

Nombre de groupes : 1 ou 2

Pitch : *Si la généralisation de la virtualisation a permis d'offrir une grande flexibilité aux applications modernes, elle s'accompagne d'une fragmentation de la mémoire. Ce projet PSAR vise à développer une solution permettant, en mode utilisateur, de partager entre deux machines virtuelles la mémoire inutilisée au travers d'un cache réparti. Le but étant de comparer les performances obtenues avec celles des solutions noyau existantes.*

Mots-clés : cache réparti, programmation système, programmation noyau, virtualisation.

Contexte : La virtualisation est aujourd'hui omniprésente dans les systèmes distribués : elle permet en effet de fournir dynamicité et sécurité. Cependant cette flexibilité se fait au prix d'une importante fragmentation de la mémoire. Or le bon dimensionnement des machines virtuelles (VMs) reste une tâche ardue et les administrateurs surdimensionnent souvent la mémoire allouée à chaque VM, générant ainsi un énorme gâchis de ressources.

L'utilisation de la mémoire *libre* par le système d'exploitation pour faire du cache règle partiellement le problème : les applications effectuant des Entrées/Sorties en sont bénéficiaires, mais si une VM utilise peu son disque, le cache sera vide ou inutile. De plus, même avec du cache, une application intensive en Entrées/Sorties se trouve fortement ralentie par le moindre accès au disque, alors que de la mémoire *libre* est peut être disponible sur une autre VM.

Une solution consiste alors à utiliser la mémoire *inutilisée* d'autres machines en réseau : la latence d'un réseau local est beaucoup plus faible qu'un accès disque, il est donc intéressant de disposer d'un second niveau de cache, réparti en réseau. Si des solutions de cache réparti en espace utilisateur existent déjà (*Memcached*), elles sont loin d'être transparentes et nécessitent de lourdes modifications des applications pour en profiter.

Objectifs : L'idée de ce projet est de concevoir et d'implémenter un cache réparti transparent pour les applications. Des solutions noyau existent, mais elles induisent un surcoût lié aux changements de contexte. Ce projet vise à développer une solution s'exécutant en mode utilisateur. Plusieurs approches système sont envisageables : surcharge des fonctions d'Entrée/Sortie Posix, réalisation d'un système de fichier utilisateur (*FUSE*), ...

Déroulement du projet :

1. Développement des primitives pour capturer les accès au système de fichiers (Posix ou FUSE) ;
2. Développement d'un cache *local* en espace utilisateur. Pour cela, il faut contourner le *page cache* du noyau (`O_DIRECT`) et implanter une stratégie de cache efficace ;
3. Test de performances du cache local en espace utilisateur comparé au *page cache* du noyau ;
4. Développement du cache réparti : les données évincées du cache *local* sont envoyées dans un cache « distant »en réseau (sockets Posix ou *MPI*) ;
5. Test de performances du cache réparti comparé au cache *local* et au *page cache* du noyau.

Les étudiants pourront avoir accès, le cas échéant, à la plateforme Grid'5000 pour réaliser l'étude de performance.

Pré-requis : Bon niveau en C et en programmation système, notions en noyau.

Liens :

— `http://fuse.sourceforge.net`
— `https://www.grid5000.fr`
— `http://www.open-mpi.org`