# TP 5 : Découverte du réseaux *Myrinet*

Programmation avec GM

Alexandre DENIS
denis@labri.fr

Systèmes et supports d'exécution — 2004–2005

L'objectif de ce TP est de découvrir le réseau haute performance *Myrinet* à travers son interface de programmation GM 2.0.

## 1 Infrastructures

Le TP a lieu en salle 007 — seule salle équipée en *Myrinet*. Les machines ont une installation locale, sans NFS. Les machines bootent par le réseau *via* PXE, sauf la machine `primus` qui sert de serveur pour toute la salle.

Pour vous connecter, utilisez le compte `guest`, mot de passe `guest`. Les disques locaux sont régulièrement régénéré. Par conséquent, **n'oubliez pas de sauvegarder votre travail par `scp` vers votre compte à la fin de la séance !** Sinon, il sera perdu ; tout ce qui est sur le compte `guest` ou ailleurs sur le disque est supposé effaçable.

En standard, les machines n'ont pas de connexion inernet vers l'extérieur, ni même vers les autres machines du cremi. Pour autoriser la connexion, ouvrez un navigateur vers n'importe quelle adresse web. La fenêtre d'authentification s'affiche. Répondez par votre login et mot de passe habituels. **Gardez cette fenêtre ouverte** pour garder la connexion internet (donc l'accès à la documentation !) active.

## 2 Présentation de *Myrinet*

*Myrinet* est un réseau haute performance basé sur des cartes de communication dotées d'un processeur programmable (le LANAI) et de mémoire embarquée. Le réseau est entièrement commuté et *full duplex*, et les communications sont fiables (sans perte). Plutôt que de programmer directement la carte *Myrinet*, nous l'utiliserons au travers de son pilote GM, actuellement en version 2.0, qui prend entièrement en charge la gestion du LANAI.

La documentation complète de GM 2.0 est disponible en ligne sur `http://www.myri.com/scs/GM-2/doc/`

Le pilote GM n'est pas chargé par défaut. Commencez par passer super-utilisateur : faites `Ctrl-Alt-F1` pour passer en console texte, login `root`, mot de passe `root`, puis chargez GM à l'aide de la commande `/opt/local/etc/init.d/gm start` À tout moment, vous pouvez vérifier que GM est effectivement chargé si le module `gm` apparaît dans la liste en tapant `lsmod`.

Vous n'avez ensuite plus besoin de la console `root`. Tous vos travaux se font sous le nom `guest`. Appuyez sur `Ctrl-Alt-F7` pour revenir à la console graphique.



## 3 Travail à faire

Commencer par lire attentivement au moins les sections 3 (*overview*) et 4 (*Programming model*) de la documentation en ligne.

### 3.1 Envoi simple

Ecrire un programme émetteur et un programme récepteur s'échangeant un message de 10 octets (taille fixe et connue par les deux parties).

Indications :

- attention à l'allocation mémoire ! Utiliser les primitives `gm_dma_*alloc` décrites à la section 6 (*Memory setup*) et non le `malloc` habituel.
- attention à l'ordre des octets. GM envoie es données avec les octets en ordre « réseau », différent de l'ordre « machine ». Consulter la section 10 (*Endian conversion*) à propos des fonctions `gm_hton_*` et `gm_ntoh`.
- ne pas chercher à faire du zéro-copie pour l'instant !
- prendre garde à la façon de spécifier les tailles de message selon GM, en particulier la distinction faite entre la notion de *size* et *length* à la section 3 (*overview*).

### 3.2 Ping-pong

Étendre le programme précédent pour réaliser un ping-pong, de façon à pouvoir mesurer les performances.

### 3.3 Benchmark

Modifier le programme de façon à envoyer des messages de taille croissante (les puissances de 2, de 4 octets à 4 Mo par exemple), et mesurer le temps de transfert pour chacune des tailles. Tracer ensuite la courbe de débit en fonction de la taille des messages à l'aide de `gnuplot`.