TP 1 – RPC

Kenneth VANHOEY
https ://dpt-info.u-strasbg.fr/∼kvanhoey [1]

Il est conseillé d'utiliser les pages de manuel via **man registerrpc** et **man xdr_string**. Regarder également les fichiers *rpc.h* et *xdr.h* du répertoire */usr/include/rpc*. Regardez également la commande *rpcinfo* **(man rpcinfo)** qui vous permettra notamment de vérifier quels services rpc sont actifs.

1 Premier exemple

Compiler puis tester l'exemple vu en TD se trouvant dans le répertoire *1-exempleTD_RPC/* :
- Utiliser la commande **makefile** pour compiler ;
- Lancer le serveur et constater qu'il est enregistré au niveau du serveur de liaison (le numéro de programme en décimale de votre service est le *536871168*) ;
- Dans une autre console, lancer le client (avec comme argument le serveur *localhost*) et observer les écritures sur la sortie standard des deux côtés. Comparer avec ce qu'on a vu en TD et déduisez-en le boutisme de la machine.

2 Fonctions d'encodage/décodage personnalisées

Compiler et tester l'exercice sur l'encodage/décodage XDR vu en TD. Le fichier source et le makefile se trouvent dans le répertoire *2-exempleTD_XDR/*.

1. Expliquez pourquoi il y a une erreur d'encodage sur *chaine0* et non sur les données précédemment encodées ;
2. Quelle est la valeur minimale à donner à *TAILLE* pour que l'encodage de la *chaine0* se passe bien ?
3. Et pour la *chaine1* ?

Remarque : vous rédigerez vos réponses à ces questions dans un fichier *reponses.txt*.

3 Division entière : RPC et filtres XDR

Définir un appel RPC qui permette de calculer, de façon distante, la division entière d'un nombre par un autre (en récupérant le quotient et le reste). Un squelette de programme est fourni dans le répertoire *3-divisionEntiere/*. Dans tous les fichiers (4 au total), il y a des « ? ? ? » à remplacer par le code correct.

Pour tester votre application distante, vous allez vous positionner sur deux machines : *turing* et *codd* (il y a un montage automatique des fichiers de l'un sur l'autre : les modifications sur l'un se répercutent immédiatement sur l'autre). Vous développerez votre code sur *Turing* et exécuterez l'un de vos deux exécutables sur *codd*. Pour cela, gardez une console ouverte en permanence sur *codd* grâce à la commande *ssh codd*.

4 Exercice complet : RPC et XDR

Pour cet exercice, chaque sous-section se fera dans un répertoire différend, respectivement *4-Matrices22/* et *4-MatricesNN/*.

1. Sujet créé à partir de documents de Guillaume LATU

4.1 Calcul de matrices 2×2

Implantez un serveur de calcul sur des matrices 2×2 à coefficients réels et offrant comme services :
- la multiplication de deux matrices ;
- l'addition de deux matrices.

Le client offrira la possibilité d'appeler le service de multiplication ou d'addition, au choix.

4.2 Calcul de matrices $N \times N$

Reprenez la section précédente considérant que les matrices sont carrées et de taille $N \times N$. Mettez en évidence qu'il existe un time-out (au niveau du client) et une retransmission lorsque l'appel de fonction distant prend un peu de temps.

Déterminez la taille d'envoi maximal d'un bloc RPC, sachant qu'il s'agit toujours d'un multiple de 1024 octets, avec un minimum de 8Ko. Quelle est alors la taille N_{max} maximale pour l'envoi d'un couple de matrices $N \times N$? Justifiez votre propos dans *reponses.txt*.

Remarque La commande **rpcgen** permet de générer des filtres XDR et de profils de fonctions automatiquement à partir d'une déscription générique. Si vous envisagez de faire votre projet en RPC, pensez à regarder cet outil.