Synthèse

1 Sujet

Réalisez un démon d'information du système, c'est-à-dire un programme qui se chargera de récupérer et traiter des requêtes. Un client pourra émettre une requête pour demander :

- Les informations concernant un utilisateur à partir de son uid ou de son nom (login, nom réel, groupe, répertoire dédié, shell utilisé, ...) ;
- Les informations sur un processus en fonction de son `pid` (commande, propriétaire, état, ...) ;
- Toute commande système usuelle.

2 Présentation

2.1 Architecture globale

La communication entre le démon et les clients devra se faire selon le schéma suivant (voir aussi Figure 1) :

- Les clients demandent l'attribution d'une ressource au démon via un SHM implémentant une file synchronisée de taille limitée ;
- Pour chaque demande reçue, le démon associe une ressource sous la forme d'un thread dédié à la gestion des requêtes du client. La communication entre le thread et le client sera alors effectuée via un couple de tubes ;
- Le client indique sa requête au thread via le premier tube. Cette requête peut comporter un nombre variable de paramètres ;
- Le thread exécute la commande, puis retourne le résultat via le second tube. Le client détecte la fin de la réponse lorsqu'il atteint la fin de fichier sur le second tube.
- Après avoir reçu la réponse, le client peut émettre une seconde requête via le premier tube, ou mettre fin à la communication en fermant son descripteur sur le premier tube.

2.2 Gestion des requêtes

Chaque requête reçue par le thread devra être traitée par un appel à une commande spécifique pour chaque demande client. Les commandes implantées seront celles que vous avez développées lors des séances de TP, dont `info_proc` et `info_user`. Ces commandes sont des programmes qui reçoivent les paramètres de la question en arguments et qui écrivent la réponse sur leur sortie standard.

2.3 Contraintes

Votre application sera composée d'un démon et d'un client répondant aux impératifs suivants :

- Le démon sera capable de gérer plusieurs clients en parallèle ;
- Le démon devra gérer correctement les zombies et les demandes de terminaisons via des signaux ;
- Le segment de mémoire partagé permettant au client de déposer ses requêtes sera un tampon de taille fixe (paramétrable), il faudra donc gérer le cas où il n'est pas possible pour un client de déposer une nouvelle requête ;
- Les fonctionnalités d'utilisation de la zone de mémoire partagée seront définies dans une bibliothèque dynamique ;
- Les éléments définis en paramètres seront enregistrés dans un fichier de configuration, et les valeurs des paramètres prises en compte au démarrage de l'application.

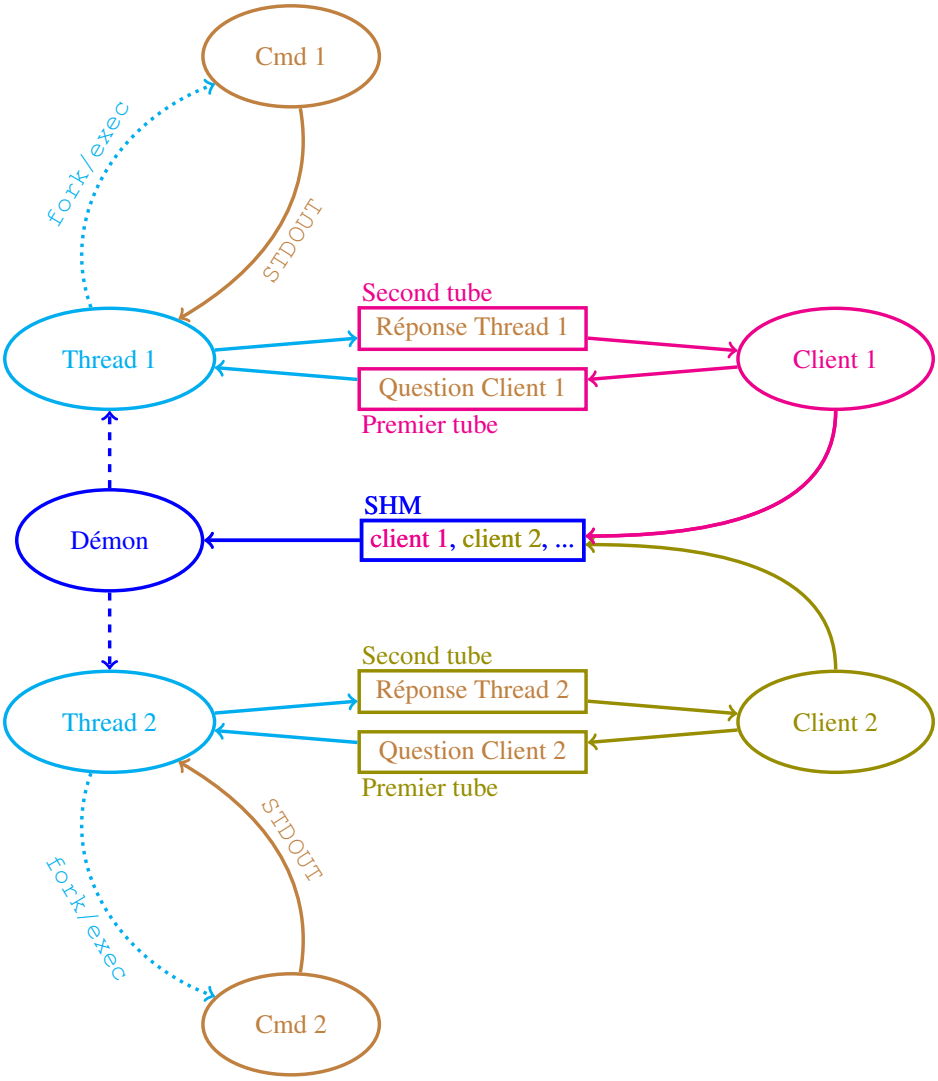

FIGURE 1 – Schéma de l'architecture client/démon

3 Suggestions d'améliorations

Vous pourrez apporter toutes les améliorations que vous estimerez pertinentes à l'application.

Une amélioration possible consiste à définir un "Timeout" comme étant le temps pendant lequel un thread attend de recevoir une requête d'un client. Si aucune requête n'est reçue pendent ce temps là, le thread met fin à la connexion.

De la même façon, on peut définir un "Timeout" coté client.

4 Travail à réaliser

En plus des codes sources correctement documentés et d'un makefile respectant les flags spécifiés en début de semestre, vous rendrez :

— Un petit manuel utilisateur explicitant comment utiliser votre application ;
— Un manuel technique décrivant les solutions que vous avez été amené.e.s à développer pour réaliser les différents modes de communication entre le démon et ses clients.

Le projet peut être réalisé seul.e ou en binôme. La date limite de rendu est fixée au Lundi 3 janvier 2022. Une soutenance sera organisée en début d'année 2022.