TP - ASR7 Programmation Concurrente

—

Synchronisation et Ordonnancement

Matthieu Moy + l'équipe ASR7 (cf. page du cours)

Printemps 2020

I Préliminaire : accès distant

Pour commencer, vu que cette année le TP se fera entièrement en distant, voici quelques informations pour vous connecter sur les machines de l'université.

Pour l'exemple, nous allons essayer de nous connecter au PC `b710l0107.univ-lyon1.fr` qui se trouve physiquement dans une salle machine (sous réserve que ce PC soit allumé). S'il ne fonctionne pas, essayez de deviner l'adresse d'un autre PC du nautibus, les noms sont toujours du type `b710` (fut un temps ou le Nautibus s'appelait « bâtiment 710 »...), la lettre « l », puis 4 chiffres.

Il est possible que toutes les machines soient éteintes quand vous ferez vos tests, auquel cas vous devrez croire cet énoncé sur parole pour la connexion à un PC.

Dans un monde sans firewall, on pourrait s'y connecter simplement avec la commande :

```
ssh votre-login-lyon1@b710l0107.univ-lyon1.fr
```

En pratique, ça ne marchera pas de l'extérieur car l'accès est bloqué pour des raisons de sécurité. Voici quelques solutions :

I.1 Le VPN

Pour accéder aux machines de l'université depuis l'extérieur du réseau, l'idéal est d'utiliser le VPN. Il faut installer le client Cisco AnyConnect, puis entrer l'adresse `sslvpn.univ-lyon1.fr` dans le champ « connect to », cliquer sur « connect » puis entrer vos identifiants Lyon 1.

Vous pouvez tester en vous connectant sur un des PCs des salles machines :

```
ssh votre-login-lyon1@b710l0107.univ-lyon1.fr
```

I.2 Rebond SSH à la main sur `linuxetu`

Vous avez accès via SSH, même de l'extérieur, à la machine `linuxetu.univ-lyon1.fr`, donc vous pouvez vous connecter d'abord a linuxetu, puis dans le shell ouvert sur linuxetu lancer une connection SSH vers votre machine virtuelle. Par exemple, si votre login Lyon 1 est p1234567, vous pouvez le faire comme ceci :

```
votre-ordinateur$ ssh p1234567@linuxetu.univ-lyon1.fr
p1234567@linuxetu.univ-lyon1.fr's password: <votre-mot-de-passe-Lyon1>
[...]
==============
Ceci est une passerelle SSH.
Votre 'home universitaire' est dans le dossier HOMELYON1.
Si vous utilisez une clé SSH, vous pouvez exécuter la commande mhome pour le monter.
==============

p1234567@linuxetu:~$ ssh p1234567@b710l0107.univ-lyon1.fr
The authenticity of host b710l0107.univ-lyon1.fr can't be established.
ECDSA key fingerprint is SHA256:/VAtIv4LXOEiKzQRNYsRPcxENUvTrsuAvK31gKp8e+s.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.75.165' (ECDSA) to the list of known hosts.
chaprot@192.168.75.165's password: <mot-de-passe-de-la-vm>
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-22-generic x86_64)
[...]
chaprot@vm-26:~$
```

L'ordinateur vous demande de vérifier l'empreinte de la clé, nous supposerons que tout est bon.

I.3 Rebond SSH automatique sur `linuxetu`

Sur la machine locale, on peut demander à faire passer automatiquement les connections par la machine `linuxetu`. Pour cela, dans le fichier `~/.ssh/config` (à créer s'il n'existe pas), ajoutez les lignes (remplacez votre-login-lyon1 bien sûr) :

```
Host !forge.univ-lyon1.fr !linuxetu.univ-lyon1.fr !*.tpr.univ-lyon1.fr *.univ-lyon1.fr
ProxyCommand ssh -N -W %h:%p votre-login-lyon1@linuxetu.univ-lyon1.fr
```

On peut lire ces lignes comme « Pour toutes les machines, à l'exception de la forge, de linuxetu, et des machines des salles TPR, dont le nom termine par univ-lyon1.fr, ouvrir une connexion avec linuxetu pour créer la connexion demandée. ». Si vous ne vous êtes pas trompés, vous devriez pouvoir ouvrir une connexion SSH avec simplement :

```
ssh votre-login-lyon1@b710l0107.univ-lyon1.fr
```

La commande devrait vous demander deux fois votre mot de passe : une fois pour `linuxetu` et une autre pour `b710l0107.univ-lyon1.fr`.

II Tunnel SSH

Mise en place du tunnel ssh (à lancer depuis votre machine de travail, remplacer `IP_VM` par l'adresse IP de votre machine virtuelle et `login` par votre login Lyon 1) :

```
ssh -L 20002:b710l0107.univ-lyon1.fr:22 votre-login-lyon1@linuxetu.univ-lyon1.fr
```

Dans un autre terminal, pour accéder à la machine :

```
ssh -p 20002 localhost
```

Modifier le numéro du port (ici 20002) au besoin !
Plus d'informations ici par exemple :
`https://blog.netapsys.fr/creer-des-tunnels-ssh/`.

III Introduction

Édition spéciale coronavirus : pas de présentiel cette année. Vous avez reçu les consignes par email pour poser vos questions par messagerie instantanée. Connectez-vous à la messagerie avant de démarrer le TP.

Pour ce TP, nous allons jouer avec les politiques d'ordonnancement temps-réel du noyau Linux (`SCHED_FIFO`, `SCHED_RR`). Activer ces politiques est potentiellement dangereux pour le système (une boucle infinie en priorité temps-réel peut figer l'ensemble du système), donc interdit pour des simples utilisateurs : vous aurez besoin de passer root pour le faire. Vous ne pourrez donc pas faire ce TP sur les machines physiques de l'université.

Nous allons utiliser l'infrastructure de *cloud computing* du département financée par la région Rhône-Alpes. C'est un ensemble de machines pilotées par le logiciel `OpenStack` avec lequel vous allez avoir un premier contact. Vous allez l'utiliser afin de créer une machine virtuelle, que vous pourrez conserver, effacer (mais perdre ainsi toutes les configurations effectuées), re-générer... La machine virtuelle (VM) créée va ici nous servir de base d'expérimentation, et dans le TP suivant d'entraînement à des manipulations d'administration systèmes que vous pourrez ensuite effectuer sur vos machines personnelles.

III.1 Créer sa machine virtuelle

Édition spéciale coronavirus : Cette partie est gardée pour mémoire, mais cette année la VM est déjà créée pour vous par votre serviteur. L'adresse IP de votre VM est disponible dans TOMUSS, colonne IP_VM. Dans toutes les commandes qui suivent, IP_VM est à remplacer par l'adresse IP en question.

Pour cela, vous devez vous connecter à l'interface d'administration :
`http://cloud-info.univ-lyon1.fr/` et utiliser le domaine `UNIV-LYON1.fr`, avec vos identifiants habituels Lyon 1. Vous faites partie du projet `OpenStack` ASR7 (si besoin, sélectionnez-le depuis le menu déroulant en haut de l'écran. Vous devez bien faire attention à ne pas créer trop de machines, ni utiliser trop de ressources.
Remarque : L'URL précédente, tout comme les VMs que vous pourrez générer, n'est pas accessible depuis *l'extérieur* de l'Université, sauf en passant par une machine relais, c'est-à-dire un proxy. Des explications pour vous y connecter sont disponibles en section II.

Vous devez créer une instance de votre machine grâce à
`Projet->Calcul->Instances->Lancer une instance`. Faites attention à :

- donner un nom reconnaissable pour votre instance ;
- utiliser l'image `ASR7` ;
- utiliser le gabarit (ensemble de ressources) `m1.tiny` ;
- ne créer qu'une seule instance ;

— Lancer la création.

La machine virtuelle va être créée (cela demande un peu de temps) et apparaître dans la liste des Instances. Elle obtiendra une **adresse IP** que vous noterez. Nous allons l'utiliser plus loin, sous le nom `VM_IP`. **Attendez** que l'`État de l'alimentation` soit `En fonctionnement` avant de pouvoir vous y connecter.

Édition spéciale coronavirus : vous pouvez reprendre la lecture ici.

On peut noter que la VM n'a qu'un CPU (VCPU, pour Virtual CPU), même si elle tourne en réalité sur un serveur physique qui en a beaucoup plus que ça. Tout se passera donc comme si nous étions sur une machine mono-cœur, ce qui simplifie un peu les choses en terme d'ordonnancement.

III.2 S'y connecter

Q.III.1) - Connectez-vous en utilisant l'utilisateur chaprot. Depuis l'intérieur de l'université, on peut utiliser SSH normalement comme ceci :

```
ssh chaprot@VM_IP
```

Si vous faites le TP de l'extérieur de l'université, alors malheureusement la machine virtuelle n'est pas accessible directement. Si vous avez installé le VPN Lyon 1, lancez-le et connectez-vous comme si vous étiez à l'extérieur de l'université. Sinon, utilisez un rebond SSH sur `linuxetu` avec l'une des méthodes indiquées ci-dessus (en utilisant l'IP de votre VM comme nom de machine au lieu de b710l0107.univ-lyon1.fr, et le login `chaprot` au lieu de votre login Lyon 1). Le mot de passe par défaut vous sera donné par votre enseignant.

Par exemple, la connexion avec rebond SSH manuel ressemble à ceci :

```
moy@moylip:~$$ ssh p1234567@linuxetu.univ-lyon1.fr
p1234567@linuxetu.univ-lyon1.fr's password: <mot-de-passe-Lyon1>
[...]
matthieu.moy@linuxetu:~$ ssh chaprot@192.168.75.165
chaprot@192.168.75.165's password: <mot-de-passe-de-chaprot>
[...]
chaprot@vm-26:~$
```

Q.III.2) - La première chose à faire est de **changer le mot de passe**, avec la commande `passwd`. Attention à ne pas l'oublier, vous en aurez besoin au prochain TP. Sauf autre configuration particulière, vous seul aurez accès à votre VM.

Q.III.3) - La commande SSH vous a donné un accès shell, en mode texte. Les outils graphiques dont vous avez probablement l'habitude ne sont donc pas disponibles. La commande SSH permet en général d'utiliser des commandes graphiques si on utilise `ssh -X` pour se connecter au serveur, mais cela ne marche pas sur linuxetu. Profitons-en pour nous familiariser avec des éditeurs de textes en mode texte : `nano` est un éditeur peu avancé mais simple à utiliser. `emacs` et `vim` sont deux éditeurs très avancées mais demandant une phase d'apprentissage relativement longue. Dans les trois cas, vous pouvez ouvrir un fichier simplement en appelant l'éditeur avec le nom du fichier en paramètre sur la ligne de commande, par exemple `nano sched.cpp`.

Q.III.4) - Récupérez maintenant le fichier `sched.cpp` sur votre VM. Une manière de faire est d'utiliser la commande `wget` (qui télécharge un fichier) depuis la VM :

`wget https://asr-lyon1.gitlabpages.inria.fr/prog-concurrente/tp5/sched.cpp`

Une autre est de télécharger le fichier sur votre PC physique, puis de l'envoyer à la VM avec une commande comme :

`rsync -av sched.cpp chaprot@VM_IP:`

(malheureusement, cette commande ne fonctionnera pas directement depuis l'extérieur de l'université sans VPN ou ajout dans le `~/.ssh/config` comme indiqué ci-dessus.

Q.III.5) - Vous pouvez passer root via la commande `sudo su`. Il est conseillé de travailler comme utilisateur normal (`chaprot`), et de ne passer root que quand c'est nécessaire, c'est à dire pour lancer l'exécutable de votre TP pour le cas qui nous intéresse. Par exemple :

```
$ g++ -std=c++11 -pthread sched.cpp  -o sched
$ sudo ./sched
$ nano sched.cpp
```

Les commandes `g++` et `nano` s'exécuteront avec l'utilisateur `chaprot`, alors que `./sched` s'exécutera avec l'utilisateur `root`.

Attention : la VM que vous avez créée pourra être supprimée sans avertissement. En fin de TP, récupérez vos données sur votre compte (par exemple avec la commande `rsync`).

IV Modification de l'ordonanceur

Dans cet exercice, vous allez utiliser explicitement l'ordonnanceur du noyau Linux. C'est assez dangereux car un processus prioritaire qui ne s'arrête pas, par définition, bloque l'ordinateur. Vous devez donc utiliser les machines virtuelles.

Le code `sched.cpp` a été préparé, il lance un certains nombre de threads qui font des calculs en affichant de temps en temps des informations. Pour le moment, le code de la fonction `change_ordonnancement()` n'est pas fait et l'ordonnancement n'est pas modifié.

Q.IV.1) - Complétez le code de cette fonction pour qu'elle change l'ordonnancement du thread qui l'appelle.

Il n'y a pas de fonctions C++ définies dans la bibliothèque standard pour manipuler l'ordonnanceur. Vous allez devoir utiliser les fonctions C suivantes pour faire ce travail :

— Retourne le numéro du thread qui l'appelle :

```
1    pthread_self();
```

— Récupérer les paramètres d'ordonnancement courant (elle vous permettra d'initialiser les variables) :

```
1    int pthread_getschedparam(pthread_t target_thread,
2                              int *politique,
3                              struct sched_param *param);
```

— modifier la politique d'ordonnancement et ses paramètres :

```
1    int pthread_setschedparam(pthread_t target_thread,
2                              int politique,
3                              const struct sched_param *param);
```

La valeur de la politique est définie dans des macros : `SCHED_FIFO`, `SCHED_RR` ou `SCHED_OTHER`.
Faites quelques expériences :

Q.IV.2) - Lancez 3 threads RR, l'un de priorité 4 et deux autres de priorité 2.

Q.IV.3) - Lancez 3 threads FIFO de priorité identique.

Q.IV.4) - Lancez 1 threads FIFO et 2 RR de priorité identique.

Q.IV.5) - Lancez 1 thread FIFO de priorité 99 et 2 autres FIFO de priorité plus faibles.

Q.IV.6) - Pouvez-vous stopper le programme pendant que des threads très prioritaires tournent ? Pourquoi [1] ?

Q.IV.7) - Si vous avez votre propre ordinateur, refaites les expériences sur ce dernier. Avez-vous les mêmes résultats ? Pourquoi ?

V Si le temps le permet, Intégration par la méthode des rectangles

V.1 Théorie

Les sommes de Riemann sont des sommes approximant des intégrales. Ainsi, si on considère $f : [a, b] \to \mathbb{R}$ une fonction partout définie sur le segment $[a, b]$; un entier $n > 0$ et une subdivision *régulière* définie pour $0 \leq k \leq n$ par :

$$x_k = a + k\frac{b-a}{n}$$

Alors la somme de Riemann est définie comme

$$S_n = \frac{b-a}{n} \sum_{k=1}^{n} f(a + k\frac{b-a}{n})$$

c-à-d

$$S_n = \sum_{k=1}^{n} (x_k - x_{k-1}) f(x_k)$$

Plus n est grand, plus [2] cette méthode des rectangles pour le calcul des intégrales donne une estimation proche de la valeur cherchée de l'intégrale

V.2 Pratique

Nous avons accès à des machines multi-cœur, et voulons en faire bon usage pour calculer des résultats d'intégration.

Q.V.1) - Avec ce qui a été vu en cours, vous devez proposer un programme qui sera capable de s'adapter au nombre de cœurs fourni par l'utilisateur en ligne de commande (de sorte que chacun soit utilisé pour calculer une partie d'intégrale). Le programme codé en C++ retournera la valeur de la somme totale calculée.
Vous pourrez utiliser des fontions à intégrer plus ou moins complexes pour vous assurer de la validité de votre code, et pour les questions suivantes.

1. Pour vous inspirer, vous pouvez regarder les valeurs des 2 variables du noyau `/proc/sys/kernel/sched_rt_runtime_us` et `/proc/sys/kernel/sched_rt_period_us`
2. moyennant la prise en compte des erreurs d'arrondi latentes...

Q.V.2) - Donnez 2 moyens de savoir combien de cœurs dispose la machine que vous utilisez actuellement.

Q.V.3) - À l'aide de la fonction `gettimeofday()`, vous mesurerez la durée du programme pour un nombre de cœurs valant 1, 2, 4, 8 et 64.
Commentez les résultats obtenus !

Q.V.4) - Vous pouvez trouver une solution en C utilisant OpenMP à l'URL `http://graal.ens-lyon.fr/~ycaniou/Teaching/1415/L3/integrale_OpenMP.c`. Quelles observations pouvez-vous faire ?