POLYTECHNIQUE MONTRÉAL

INF8480 - Systèmes répartis et infonuagique

TP2 - Appels de méthodes à distance

Chargés de laboratoire :
Daniel Capelo Borges daniel.capelo@polymtl.ca
Pierre-Frederick Denys pierre-frederick.denys@polymtl.ca

Automne 2019 - V3.1

1 Introduction

1.1 Prérequis

— **Intergiciels et objets répartis, Communication inter-processus, Messages de groupes :** Sun RPC, gRPC, CORBA, Java RMI et .NET Remoting.

1.2 But du TP

— Introduction à gRPC
— Introduction au calcul réparti et à la répartition de tâches
— Analyse de performance des systèmes répartis à l'aide du traçage

Le TP comporte deux parties indépendantes, il est vivement conseillé que les deux membres du binôme travaillent et comprennent les concepts abordés dans les deux parties du TP.

2 Partie 1 : Implémentation d'une application répartie

2.1 Mise en situation

Vous travaillez dans un laboratoire de recherche, et vous êtes responsable de la baie de serveur disponible pour votre groupe. Vous disposez de plusieurs serveurs, formant une grappe de calcul. Tous les membres du laboratoire ont besoin d'effectuer des tâches de calcul durant plusieurs heures.

Vous souhaitez donc implémenter un gestionnaire qui permet aux clients de soumettre leur tâche dans une file d'attente FIFO, et de décider quel serveur doit effectuer la tâche. Un serveur ne peut effectuer qu'une tâche à la fois. Vous devez utiliser gRPC (gRPC Remote Procedure Call), un système libre d'appel de procédure à distance développé par Google pour implémenter votre gestionnaire.

Dans ce TP, vous vous contenterez d'implémenter un gestionnaire qui envoie un appel gRPC sur un seul serveur. Vous devrez utiliser le traçage système (avec LTTng) afin de visualiser l'exécution des appels gRPC. Pour cela, le code a été instrumenté.

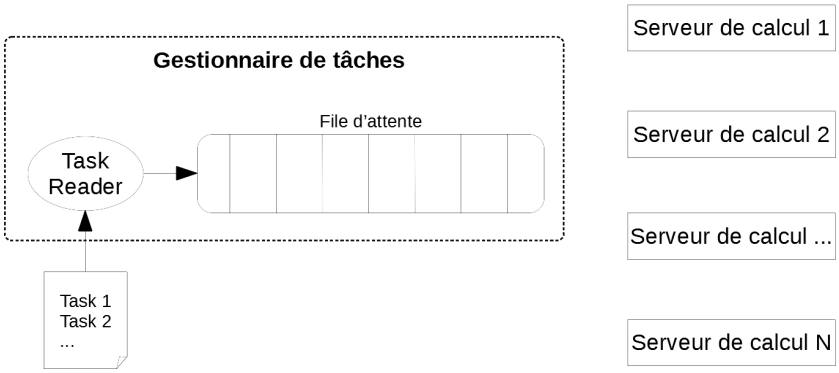

FIGURE 1 – Architecture

2.2 Mise en place

Avant de commencer vous devriez lire attentivement le paragraphes précédents, afin de dresser une liste des éléments à mettre en place et dans quel ordre. Vous devez comprendre ou chercher le rôle de chaque élément.

Un code imcomplet en C++ de l'application vous est fourni :

- `task_scheduler`
  - `lttng-traces`
    - `grpc_tracing.h` : déclaration des tracepoints
    - `grpc_tracing.c`
  - `manager.cc` : code du manager, compléter les **TODO**
  - `server.cc` : code du serveur, rien a modifier.
  - `operation.proto` : fichier d'IDL protocol buffers à compléter
  - `Makefile` : compile le serveur et le manager
  - `trace.sh` : lance le serveur en tâche de fond et la session lttng

Vous devez le compléter la déclaration des interfaces IDL, et le code du manager, qui doit prendre en paramètre le nom de la tache (`./manager tache45`). Vous pouvez éditer le code sur votre machine de labo ou personnelle, mais la compilation doit se faire sur la machine virtuelle (voir paragraphe suivant).

2.3 Tests

> **Remarque**
>
> Vous devez créer une nouvelle machine virtuelle **avec l'image** `Admin-INF8480-A2019-TP2-V3` (voir annexe), pensez à arrêter et supprimer les machines du TP1. La création de l'instance prend du temps (environ 15 min).

Démarche de test :

- Téléverser les fichiers sur une machine virtuelle Openstack que vous devez créer (voir annexe)
- Compilez votre trace provider dans le dossier `lttng-traces` :

```
gcc -I. -c grpc_tracing.c
```

- Compiler le programme avec `make` (un `Makefile` vous permet de compiler les deux programmes manager et server)
- Lancer le script `sudo sh trace.sh` (lance le traçage et le programme `server` en tâche de fond). Le script `trace.sh` permet de générer une trace système du fonctionnement de votre programme.
- Envoyez plusieurs tâches au manager : (`./manager tache23`, `./manager tache45`). Le fichier de trace en sortie est situé dans le dossier `trace_files`.
- N'oubliez pas d'exécuter les commandes suivantes pour terminer le traçage :

```
lttng stop
lttng destroy
```

— Téléverser votre dossier de trace (le dossier contenu dans `trace_files`) sur votre ordinateur de labo.
— Utilisez le logiciel `Trace Compass` (a télécharger içi : `https://www.eclipse.org/tracecompass/`) afin de visualiser la trace.

2.4 Vérification et remise

Un quiz moodle vous permet de déposer vos résultats :

1. Déposer une archive contenant le code de votre application modifiée
2. Déposer une capture d'écran de Trace Compass où l'on voit la trace des messages gPRC.

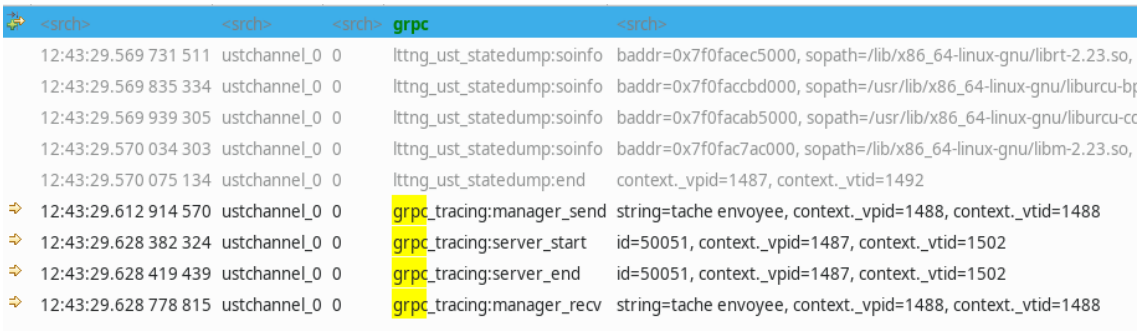

FIGURE 2 – Exemple de capture à effectuer

3. **Répondez à la question suivante :** Quels éléments devez vous ajouter/modifier afin de travailler avec plusieurs serveurs et un manager ? Dans le code des serveurs ? Dans le code du manager ?

3 Partie 2 : Analyse d'une trace système d'un système distribué

3.1 Mise en situation

Vous êtes analyste au service informatique de Poly, et on vous a signalé que le logiciel des ressources humaines était lent.

Le site est un système distribué, composé de trois parties (application cliente, serveur d'application et serveur d'authentification) qui sont des applications gRPC. Lors de la connexion, le client envoie une requête au serveur d'application, et ce dernier vérifie l'identité du client auprès du serveur d'authentification.

Vous devez donc trouver la cause et la durée des goulots d'étranglement. Les goulots d'étranglement peuvent être dans le client, le serveur, ou le serveur d'authentification.

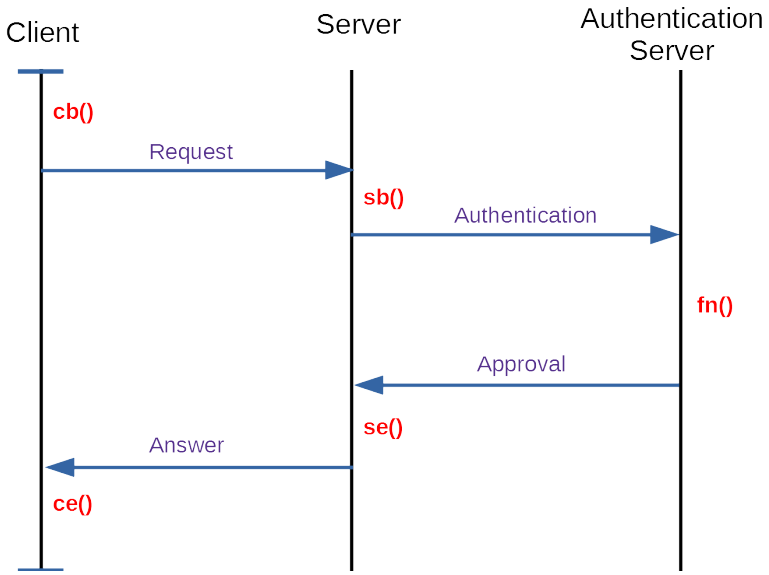

FIGURE 3 – Appel des fonctions

3.2 Analyse

Vous devez analyser à l'aide de trace compass la trace fournie `1568729535905`. Vous devez calculer le temps passé dans le client, dans le serveur et dans le serveur d'authentification. Attention, le temps d'exécution réel du client n'est pas de `client_start` à `client_end` mais de `client_start` à `server_start` et de `server_end` à `client_end`. Même chose pour le temps d'exécution du serveur.

Vous pouvez voir le delta de temps entre deux évènements avec trace compass, en sélectionnant deux évènements, avec les touches `Ctrl + Maj` enfoncées.

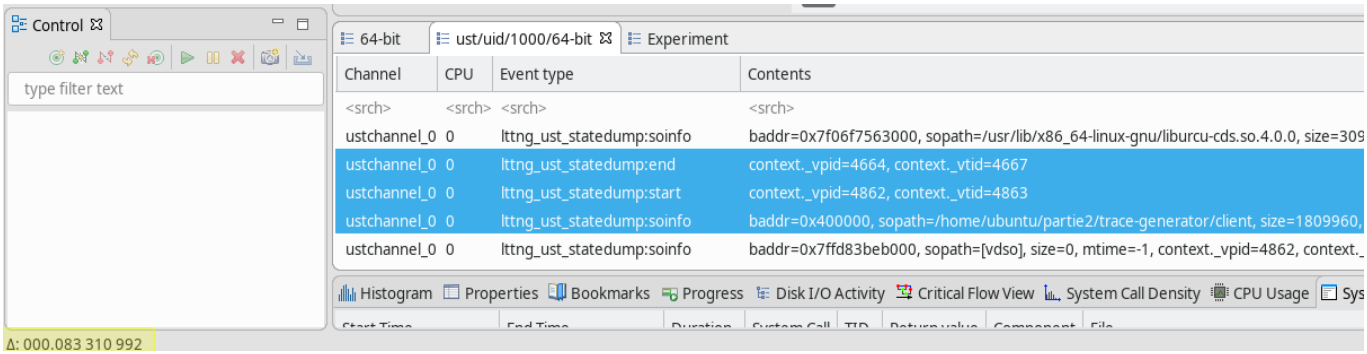

FIGURE 4 – Voir le delta de temps entre deux évènements (en bas à gauche)

3.3 Vérification et remise

Vous devez analyser la trace fournie, et répondre aux questions du quiz Moodle proposé sur le site du cours.

4 Annexes

4.1 Traçage système

4.1.1 Définition

Une trace d'un programme est une représentation de l'exécution de ce même programme. Le but est d'instrumenter et d'optimiser la qualité des programmes en termes de performances et de robustesse.

4.1.2 LTTng

LTTng est un outil de traçage et de visualisation des évènements produits à la fois par le noyau linux et par les applications.

— **Définition :** `https://lttng.org/docs/v2.10/#doc-what-is-tracing`
— **Tracepoint provider :** `https://lttng.org/docs/v2.10/#doc-tracepoint-provider`

4.1.3 Trace Compass

Trace compass est un outil de visualisation graphique de traces systèmes.

Pour ouvrir une trace, cliquez sur *File* → *Trace Import*, cliquer sur *Browse...* et sélectionner le dossier de la trace.

Puis cocher la case du dossier de la trace à ouvrir, et enfin cliquer sur finish. Le menu à gauche, permet de naviguer entre les traces systèmes et traces applications (ust). Dans ce tp, vous analyserez les traces ust (userspace).

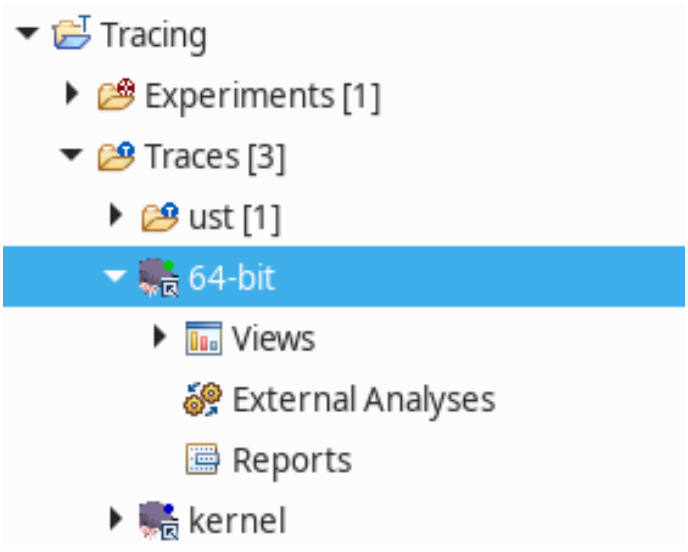

FIGURE 5 – Ouvrir la trace userspace

Vous pouvez le télécharger ici : `https://www.eclipse.org/tracecompass/`

4.2 Création d'une machine virtuelle

La création d'une nouvelle machine virtuelle peut se faire directement depuis l'interface web de Openstack (*Compute* → *Instances* → *Lancer une instance*). Il faut choisir une image et un gabarit. Dans ce TP, on va utiliser l'image `Admin-INF8480-A2019-TP2-V3` avec le gabarit `2048-20-1-1` (Ram : **2 Go**, Disque : 20 Go, Nombre des VCPUS : 1).

Toutes les instances créées pour ce travail pratique doivent être connectées au réseau `switch1-nat`. Pour accéder à une machine virtuelle, il faut la configurer avec une paire de clés SSH au moment de la création. Vous avez le choix entre créer une nouvelle paire de clés (`Paires de clés` → `Créer une paire de clés`) ou importer une existante (créée avec la commande `ssh-keygen` sur votre machine locale).

— Pour se connecter à une machine virtuelle :

```
ssh -i (cle privee) ubuntu@ip-flottante
```

— Pour envoyer un fichier vers une machine virtuelle :

```
scp -i (cle privee) nom_du_fichier ubuntu@ip-flottante:(
   repertoire_destination)
```

4.3 Associer une IP flottante à une machine virtuelle

Il est possible d'associer une adresse ip flottante à une machine virtuelle directement depuis l'interface web d'Horizon (*Réseau* → *IP flottantes* → *Associer*). Il est également possible de faire cette association depuis la ligne de la commande en suivant ces étapes :

— Récupérer le fichier rc dans le dashboard, dans *Projet* → *Accès et sécurité* → Accès API et l'exécuter :

```
source ./[LOGIN]-projet-openrc.sh
```

— S'allouer une adresse ip flottante externe (Il faut que l'instance soit connectée dans un sous-réseau interconnecté avec le réseau externe `externe1`, comme « `switch1-nat` ») :

```
openstack floating ip create externe1
```

— Ou vérifier celle qu'on a si on se l'est déjà allouée :

```
openstack floating ip list
```

— Associer notre ip externe à une instance :

```
openstack server add floating ip [name or ID instance] [
   IP_flottante]
```

♢