TP2 - Services distribués et gestion des pannes

INF4410 - Systèmes répartis et infonuagique

Nom de l'étudiant : Félix La Rocque Carrier
Matricule : 1621348

Nom de l'étudiant : Mathieu Gamache
Matricule : 1626377

Polytechnique Montréal
24 octobre 2016

Test de performances :

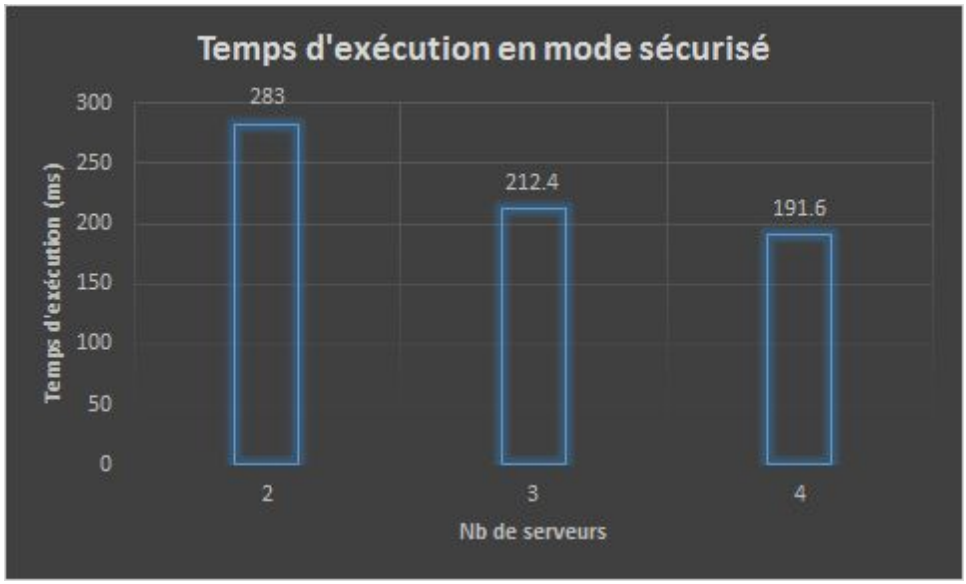
Mode Sécurisé

Caractéristiques :
2 serveurs (q = 3; q = 6)
3 serveurs (q = 3; q = 6; q = 9)
4 serveurs (q = 3; q = 6; q = 9 ; q = 12)

Avec ce type de caractéristiques de serveurs, notre implémentation devrait répondre de plus en plus vite, plus il y a de serveurs. Puisque toutes les réponses sont considérés comme valide, nous avons pas de logique de deuxième validation. En plus, avec des serveurs plus puissants (plus il y a de serveurs) nous avons une plus grande chance que le serveur réponde positivement à la tâche donner et ne soit diviser en plus petites tâches.

Puisque notre implémentation commence avec des tâches de 100 opérations et les divise en 2 parties lorsqu'un serveur répond négativement, nous avons plus de chance que les serveurs plus puissant (9, 12) répondent positivement avant les plus faibles (3, 6).

Comme on peut le voir dans le graphique des temps suivant, notre estimation est vérifié. Plus le nombre de serveur grandit, plus le temps d'exécution est rapide.



*Données prises en moyenne sur 10 exécutions

Mode non-Sécure

caractéristiques:
3 serveurs avec q=5

- Trois serveurs serveurs de bonne foi.
- Un serveur malicieux 50% du temps, deux autres serveurs de bonne foi.
- Deux serveurs malicieux 50% du temps, un autre serveur de bonne foi.

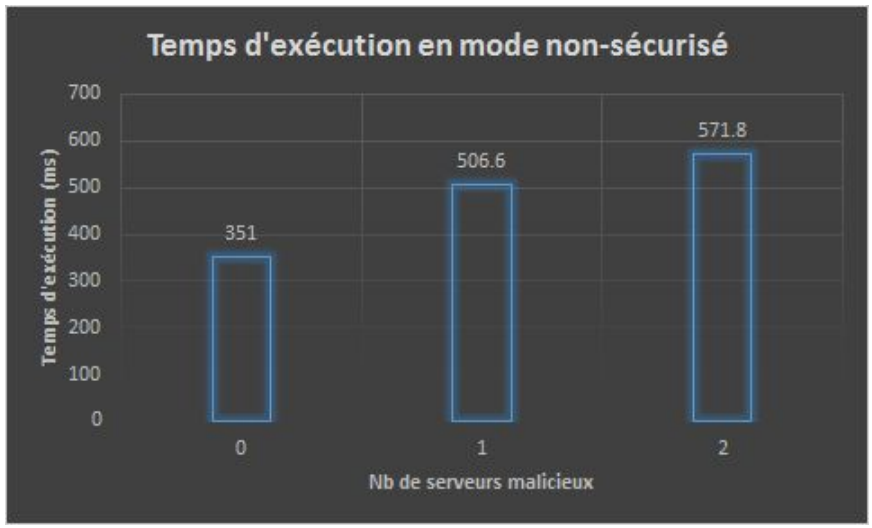
Avec ces caractéristiques, nous ne varions plus le nombre de serveurs. Ce faisant, notre stratégie de division des ressources ne causera aucune différence entre les itérations avec des caractéristiques différentes.

Dans ce cas, la stratégie de validation de réponse jouera un grand rôle dans les résultats. La stratégie adopté est dans le but d'obtenir une double validation de la part de 2 serveurs distincts. Nous avons choisie de garder une liste de résultats obtenues pour chaque bloc d'opérations (division de toutes les opérations à effectuer) et de valider un bloc de cet liste une fois que 2 réponses ont été obtenues pour celui-ci.
Si les deux réponses sont identiques, le résultat est ajouté à la somme globale et le bloc est enlevé de la file de traitement. Si les deux réponses sont différentes, la liste des réponses est vidé, ainsi que la liste des serveurs ayant traiter la requête, et on remet le bloc dans la liste des bloc à traiter.

Concurrence:
Nous avons aussi à gérer l'accès à cette liste puisque l'exécution se fait sur plusieurs threads. Dans le cas ou un résultat est obtenue pour un bloc ayant été séparée (autre serveur a répondu négativement), le résultat est ignoré et une autre tâche est attribué au serveur.

Avec cet stratégie, plus nous avons de serveurs malicieux, et plus ceux-ci ont un taux élevé de mauvaises réponses, plus le traitement des opérations sera lente.

Les données suivantes valident donc la stratégie.



Comme nous pouvons le voir, plus le nombre de serveur malicieux est grand, plus le temps requis pour compléter le calcul est grand.

Questions

Question 1: Le système distribué tel que présenté dans cet énoncé devrait être résilient aux pannes des serveurs de calcul. Cependant, le répartiteur demeure un maillon faible. Présentez une architecture qui permette d'améliorer la résilience du répartiteur. Quels sont les avantages et les inconvénients de votre solution? Quels sont les scénarios qui causeraient quand même une panne du système?

Une solution pour le rendre plus résilient serait de mettre en cache les résultats connues (de mi-calcul) de temps en temps pour un long calcul. Le serveur pourrais prendre un peu de temps pour mettre en mémoire permanente le résultat d'une partie du calcul ainsi que le reste des calculs à faire. Si celui-ci plante en plein milieu d'un calcul, il peut interroger la mémoire pour voir si il y avait un calcul en cours et où il était rendu.
Cela permet de garder un résultat partiel et de s'épargner une partie des calculs déjà fait mais cela prend des opérations supplémentaires d'écriture sur disque et une logique de reprise de calcul en cas de panne.

Une panne peut survenir si le disque dur du répartiteur est hors d'usage ou encore si le service est retiré du réseau d'une quelconque façon.