Systèmes d'exploitation
Programmation réseaux

TP 5 - Mise en place d'une architecture client-serveur Java classique avec sécurité logicielle hybride symétrique et asymétrique.

Liège, Avril 2015
HEPL - Ingénieur Industriel

Thibault Reinders
Nassim Rahali
Kaoutare Ahallouch

M 18

Professeur : C. Vilvens

Table des matières

1 Méthodologique de travail

Nous nous sommes tout d'abord réparti le travail selon les 3 serveurs à faire (Bank, Data et Web).

- Bank Server : Thibault
- Data Server : Kaoutare
- Web Server : Nassim

Ensuite, en fonction de l'avancement de chacun dans son serveur, nous avons fait le reste du travail comme suit

- Kerberos (AS & TGS) : Nassim et Thibault
- Bank Client : Thibault
- Agent Bancaire : Kaoutare et Thibault (chacun la partie utilisant son serveur.)

Pour faire les 3 serveurs, nous n'avons pas eut besoin de nous mettre d'accord puisqu'il n'y a pas d'interactions directes entre les différents serveurs, hormis le Web Server qui devait avoir un accès direct aux bases de données des deux autres serveurs.

Pour l'Agent Bancaire, les interactions entre la partie utilisant le Bank Server et celle utilisant le Data Server n'est pas très importante et nous n'avons pas eut de problème pour développer les 2 en parallèle.

Pour l'AS et le TGS, il nous a fallu travailler ensemble et communication par communication pour ne pas se tromper dans qui chiffre quoi et avec quelle clé.

2 Bank Server

Le serveur a 2 tâches à remplir. Il communique avec les clients Bank Client et Agent Bancaire. On utilise donc 2 *ThreadPoolExecutor* pour qu'un trop grand nombre de clients d'un type ne bloque pas l'autre type de client. Logiquement, on donnera plus de threads au pool destiné à communiquer avec les Banques Clientes qu'à celui pour les Agents Bancaires.

Le serveur a accès a une base de données *MongoDB* qui contient les banques clientes et leurs demandes d'opérations de crédit et de débit avec notre banque. Typiquement

```
{
  "_id" : 1,
  "type" : "debit",
  "montant" : 1000,
  "valide" : "false",
  "date" : "2015-04-15"
  "banque" :
  {
    "_id" : 1,
    "name" : "Belfius"
  }
}
```

La communication avec les Banques Clientes se fait par SSL, on a donc créé des clés (vérifiées par un CA) enregistrées dans un keystore, côté client et côté serveur. Pour ne pas alourdir le code, la création d'une socket SSL a été mise dans un petit jar.

Par défaut, les demandes de crédit ou de débit des Banques Clientes sont à *false*. Il faudra l'intervention d'un Agent Bancaire pour la valider.

L'Agent Bancaire peut demander à voir toutes les opérations ou alors les filtrer selon la Banque ou le type d'opération, pour ensuite en sélectionner une ou plusieurs pour les valider.

3 Bank Client

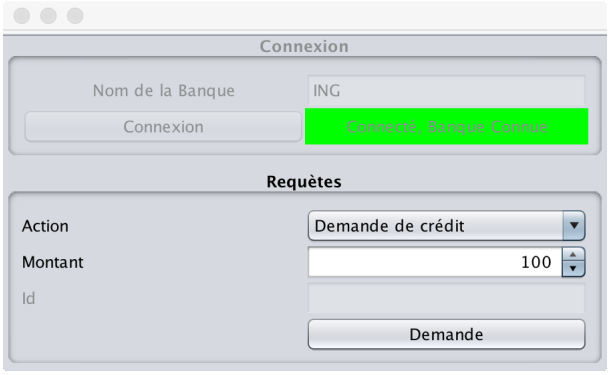

FIGURE 1 – Application Banque Cliente

4 Agent Bancaire

4.1 Vers Bank Server

4.2 Vers Data Server

5 Data Server

6 Web Server

7 AS & TGS