Modal Réseaux –- Jeu multijoueur : Love Letter Game

Quei-An CHEN, Chia-Man HUNG

Introduction

L'objectif de ce projet est de réaliser un jeu multijoueur en s'appuyant sur les connaissances de réseaux acquises au cours des amphis, à savoir la transmission des messages à travers UDP et TCP et la cryptographie. Nous avons implémeté Jframe pour réaliser l'interface graphique en Java.

Plan

- Règles du jeu
- Architecture générale
- Spécification
- Amélioration

Règles du jeu

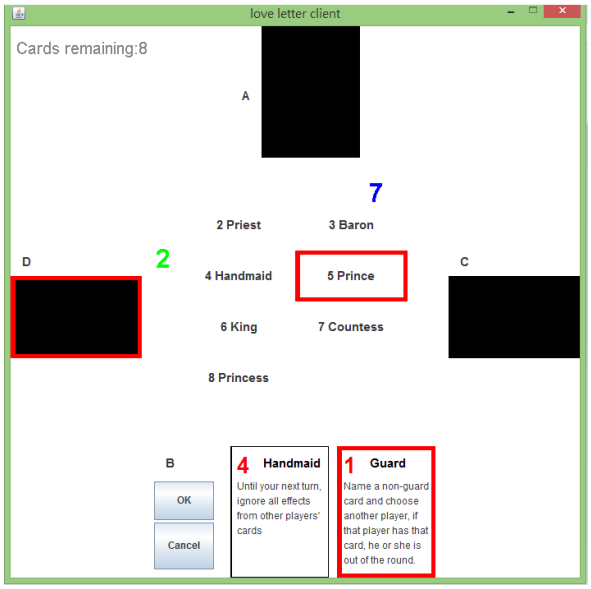
Le jeu comporte 16 cartes dont une est écartée lors d'un jeu à quatres joueurs, les cartes sont : 5 guards (1 point), 2 priests (2 points), 2 barons (3 points), 2 handmaids (4 points), 2 princes (5 points), 1 king (6 points), 1 countess (7 points), 1 princess (8 points). Chaque carte possède son unique fonction (par exemple le priest permet de regarder la carte d'un autre joueur).

Chaque joueur a une carte quand le jeu commence, le premier joueur qui joue est celui qui se trouve au sud sur le panneau du GameRoom, il tire une carte sur le deck, et il décide de jouer l'une des deux cartes qu'il a (avec sa carte initiale), et le jeu continue dans le sens contraire des aiguilles d'une montre. Les cartes jouées précédemment sont toutes visibles pour tous les joueurs (mais la carte de chaque joueur reste secret bien sûr).

Au cours du jeu, certains joueurs pourraient se trouver dans l'état « out », c'est qu'ils sont attaqués par les autres joueurs et qu'ils perdent alors totalement le droit de jouer dans le jeu.

Si l'une des conditions suivantes est satisfaite, le jeu termine :

1. Le deck s'épuise, c'est alors le joueur qui a le point sur la carte le plus élevé (et qui n'est pas « out ») gagne, s'il y en a plusieur, ils sont tous gagnants.
2. Tous les joueurs sont « out » sauf un, alors ce dernier gagne.



Architecture générale

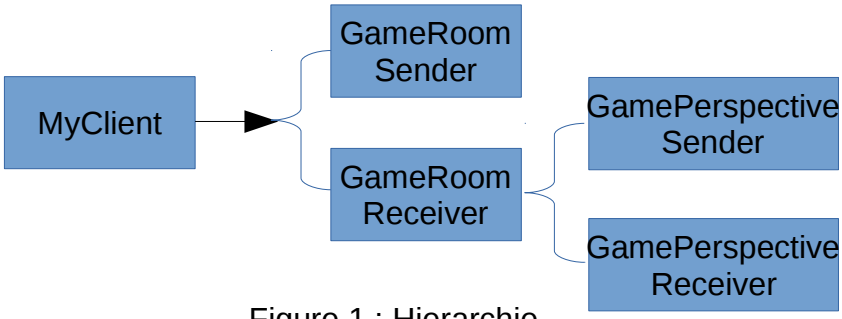


Figure 1 : Hierarchie

Pour commencer un jeu, le serveur doit d'abord être lancé. Ensuite, nous exécutons la classe MyClient. Les clients doivent savoir l'adresse ip du serveur.

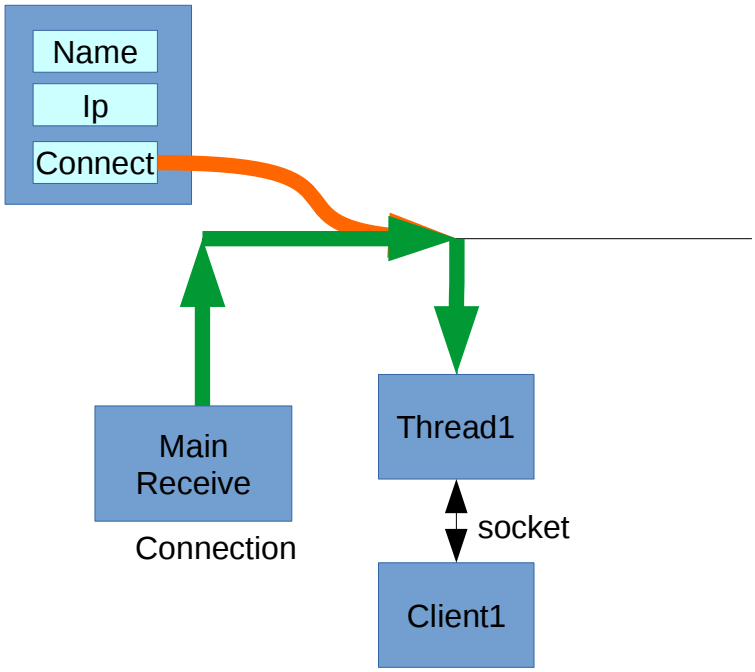


Figure 2 : MyClient

Dans la fenêtre de MyClient - Le joueur choisit un nom et l'adresse ip du serveur, si la configuration est bonne, lorsqu'il appuie sur le bouton « Connect », la connexion entre le client et le serveur s'effectue (avec le protocole TCP), puis cette fenêtre ferme et une fenêtre de GameRoom apparaît.

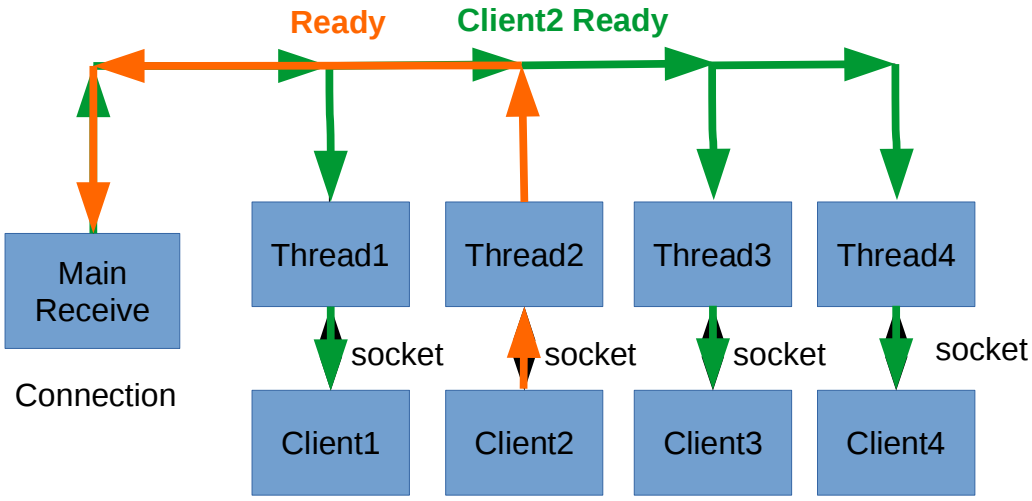


Figure 3 : GameRoom – Transmission d'état

Dans la fenêtre de GameRoom - Lorsqu'un client est connecté la première fois au serveur, ils échangent leur clés publiques d'abord un à un, puis le serveur envoie les informations des autres joueurs à ce client (leur position, nom, prêt ou non). Ensuite, quand un joueur modifie son état (sa position, prêt ou non, ou même il quitte le jeu), il envoie un message au serveur, et le serveur broadcaste ce message à tous les joueuer qui sont encore connectés.

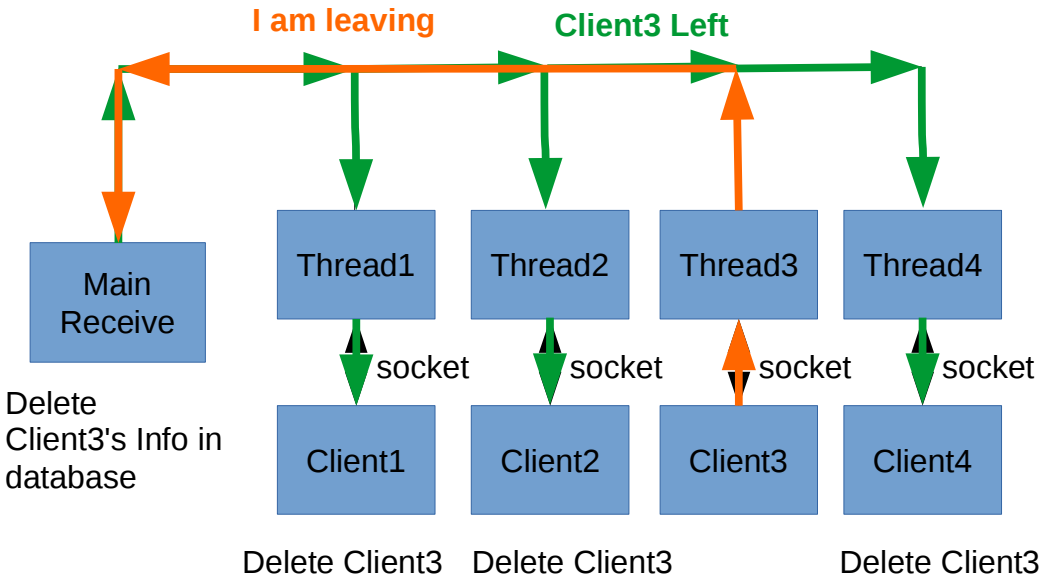


Figure 4 : GameRoom - Exit forcé

Lorsqu'un client ferme sa fenêtre, avant d'arrêter le programme, le client envoie un message de départ au serveur, alors le serveur supprime ses données dans la mémoire, puis renvoie ce message de départ à tous les autres joueurs.

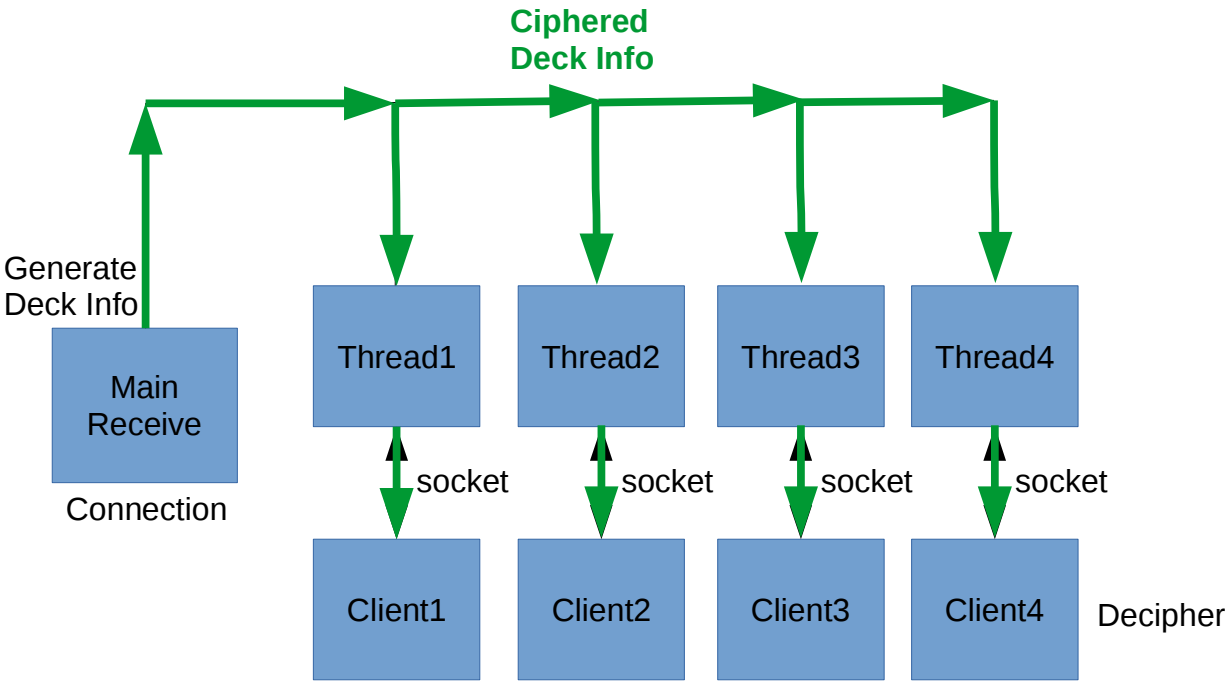


Figure 5 : GameRoom - Cryptographie

Finalement, quand tous les joueurs sont prêts, le serveur envoie un message « start » et l'information totale (les noms des joueurs, la position relatives des joueurs, les adresses ip et les ports disponibles, et le deck) sous forme cryptée avec les clés publiques des clients respectivement à tous les clients. Le serveur se ferme alors et les fenêtres de GamePerspective apparaissent.

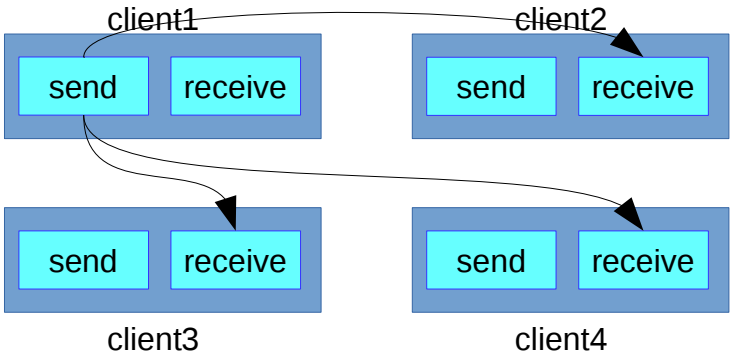


Figure 6 : GamePerspective

Dans la fenêtre de GamePerspective - Les codes du jeu se trouvent ici. Dans cette phase, chaque client ouvrent deux ports, l'un pour envoyer et l'autre pour recevoir. Lorsqu'un joueur choisit un coup légal et appuie sur le bouton « OK », il envoie un message aux trois autres joueurs avec le protocole UDP (les destinations étant déjà fournies par le serveur auparavant), et chaque joueur, quand il reçoit un message, le traite sur son propre ordinateur.
Lorsque le jeu est terminé, les joueurs peuvent soit quitter soit cliquer sur « Restart ». La dernière option permet d'accéder directement à GameRoom sans devoir retaper le nom et l'ip.

Spécification

Terminologie
Joueur courant : le joueur qui a le droit de jouer à ce tour.
Joueur ennemi : la cible du joueur courant.
Character choisi : la carte jouée par le joueur courant, noté par un entier entre 1 et 8.
Ennemi character choisi : la carte que le joueur courant devine que le joueur ennemi possède, noté par un entier entre 2 et 8.
Phase GameRoom : les joueurs sont en train de choisir leur position, et d'attendre que tous les joueurs soient prêts.
Phase GamePerspective : les joueurs jouent chacun son tour dans le sens contraire des aiguilles d'une montre.

Format des messages
Pour l'envoi des informations crytographiques (les clés, le vecteur initial pour le AES), nous envoyons d'abord un entier qui est la longueur du tableau contenant l'information, puis nous envoyons ce dernier.
Lors de la phase GameRoom, les informations echangées sont de forme « nom : position : entré ou non : prêt ou non ».
Lors de la phase GamePerspective, les informations echangées sont de forme « nom du joueur courant : nom du joueur ennemi : character choisi : ennemi character choisi ».

Détails des protocoles
Dans la phase GameRoom – TCP : le serveur ouvre son port 5000, les quatre clients connectent avec un port disponible quelconque noté x_i (i entre 1 et 4).
Dans la phase GamePerspective – UDP : les clients utilisent le port x_i pour envoyer les messages et le port x_i+100 pour recevoir les messages.

Détails de la cryptographie
Nous avons repris les classes JSRB, Alice et Bob données comme exemple dans le cinquième cours et nous avons ajouté quelques méthodes dans JSRB.

```
static byte[] receiveByte(InputStream is, int size)
static int receiveInt(InputStream is)
static void sendInt(PrintStream writer, int length)
```

Ces méthodes sont utilisées pour faciliter la transmission des messages entre le serveur et les clients, notamment pour les échanges de clés publiques et le vecteur initail qui sont sous forme de tableau de bits.
Nous avons limité la longueur de clé plublique et de clé privée à 128 bits pour ne pas avoir de `java.security.InvalidKeyException: Illegal key size`.

Amélioration

Pour l'instant nous n'avons pas traité le cas où le joueur se déconnecte au cours d'un jeu déjà commencé. Pour gérer ce problème, nous avons pensé à envoyer un message à tous les autres joueurs lorsqu'un joueur quitte le GamePerspective. Ensuite, à son tour, le programme joue automatiquement une carte suivant une règle fixée. Par exemple, il joue la carte que le joueur a tiré du deck. Si cela provoque une incompatibilité, il joue l'autre carte. S'il doit choisir un autre joueur, il choisit toujours le jouer suivant. S'il doit choisir un numéro, il choisit 2. Ceci permettrait aux autres joueurs de continuer le jeu.

La partie de l'interface graphique est également à améliorer. Cependent, ce n'est pas l'objectif de ce projet. On s'est donc contenté de ce que nous avons réalisé.