

# Rapport du Projet SH13 – Module Systèmes d'exploitation

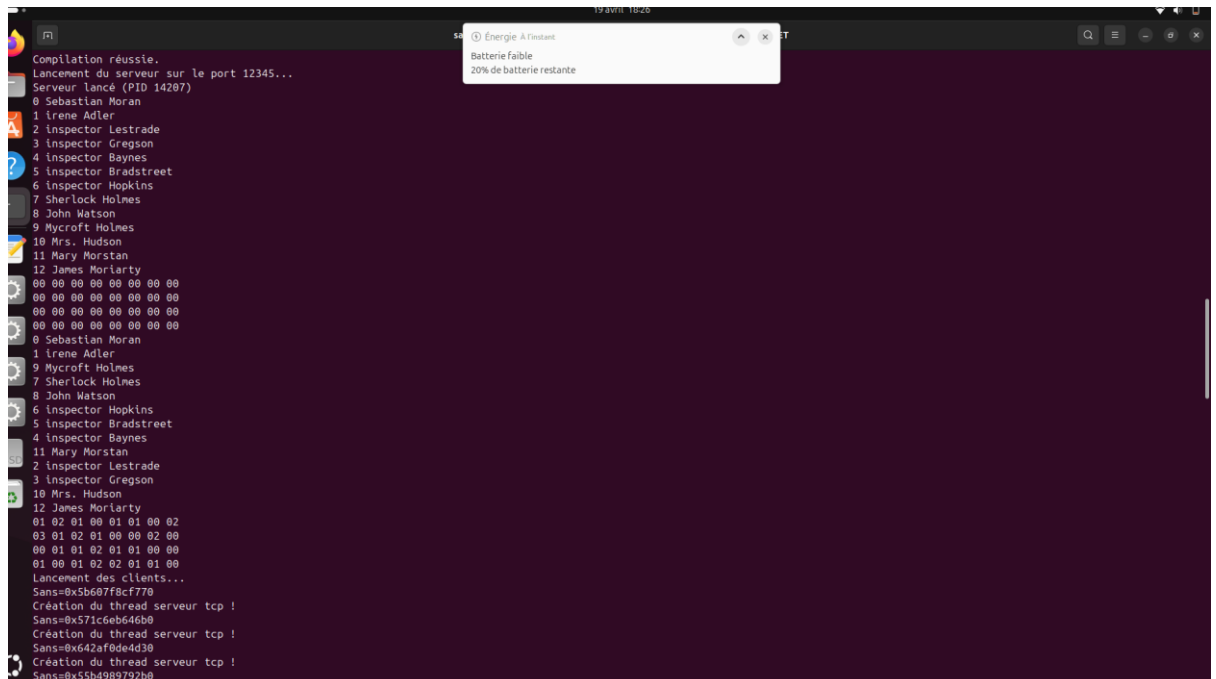
## Objectif du projet

Ce projet consistait à réaliser un jeu de cartes multijoueur inspiré du jeu Sherlock 13 (SH13). Il a été développé en langage C avec une interface graphique SDL2, et repose sur une architecture client/serveur utilisant les sockets TCP pour la communication. L'objectif principal était d'appliquer concrètement les concepts abordés en cours de Systèmes d'exploitation : sockets, processus, threads...

## Construction du programme

### Structure globale

- server.c : gère la logique du jeu, la distribution des cartes et les communications avec les clients.
- sh13.c : code client SDL, permet d'interagir avec le joueur et de recevoir/envoyer des messages au serveur.
- cmd.sh : script Bash pour compiler et lancer automatiquement serveur et clients.



```
Compilation réussie.
Lancement du serveur sur le port 12345...
Serveur lancé (PID 14207)
0 Sebastian Moran
1 Irene Adler
2 Inspector Lestrade
3 Inspector Gregson
4 Inspector Baynes
5 Inspector Bradstreet
6 Inspector Hopkins
7 Sherlock Holmes
8 John Watson
9 Mycroft Holmes
10 Mrs. Hudson
11 Mary Morstan
12 James Moriarty
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
0 Sebastian Moran
1 Irene Adler
9 Mycroft Holmes
7 Sherlock Holmes
8 John Watson
6 Inspector Hopkins
5 Inspector Bradstreet
4 Inspector Baynes
11 Mary Morstan
2 Inspector Lestrade
3 Inspector Gregson
10 Mrs. Hudson
12 James Moriarty
01 02 01 00 01 01 00 02
03 01 02 01 00 00 02 00
00 01 01 02 01 01 00 00
01 00 01 02 02 01 01 00
Lancement des clients...
Sans=0x5b607f8cf770
Création du thread serveur tcp !
Sans=0x571c6eb646b0
Création du thread serveur tcp !
Sans=0x642af0e4d30
Création du thread serveur tcp !
Sans=0x55b4989792b0
```

- Makefile : fichier de compilation alternatif à cmd.sh.

# Modifications et complétions

## Serveur

- Lecture et traitement des messages 'C' (connexion), 'G' (accusation), 'O' et 'S' (interrogations).
- Broadcast des informations 'L', 'D', 'M', 'V', 'WIN', 'LOSS' selon les actions.
- Distribution aléatoire des cartes avec `melangerDeck()` et affectation des symboles avec `createTable()`.
- Synchronisation du début de partie une fois les 4 joueurs connectés.

## Client

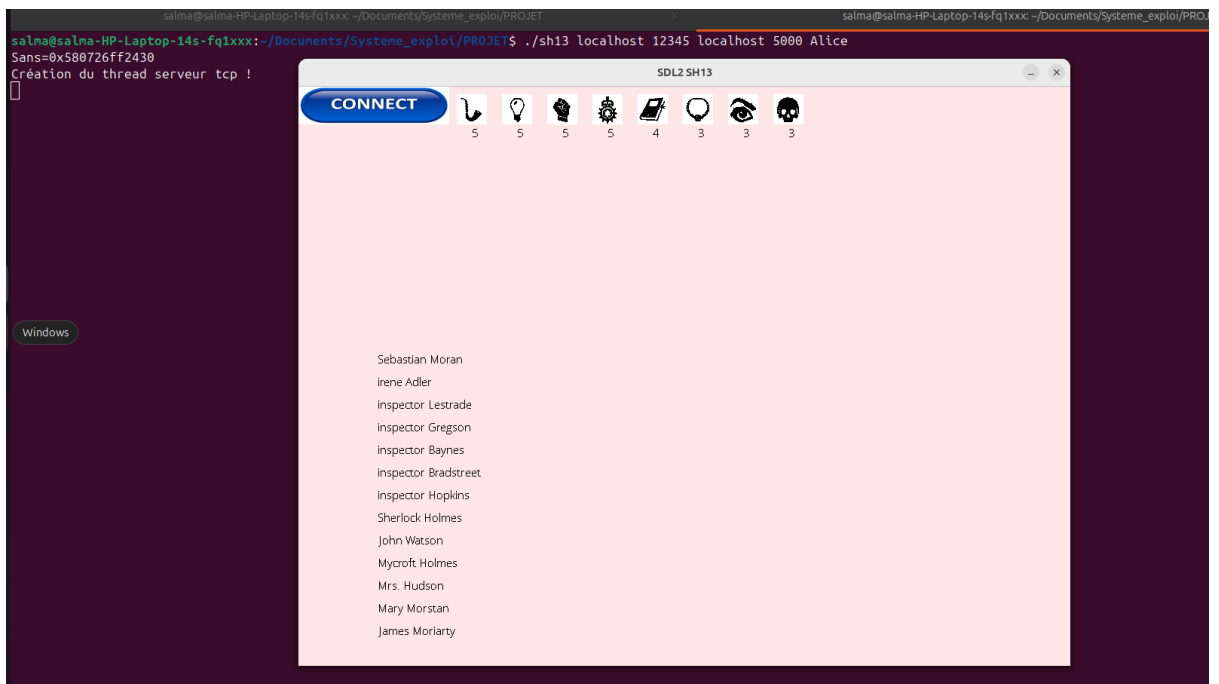
- Traitement SDL des clics souris selon les zones (objets, joueurs, cartes...)
- Communication avec le serveur via `sendMessageToServer()`
- Création d'un thread TCP pour recevoir les messages sans bloquer l'interface.
- Décodage et affichage des messages : 'I', 'L', 'D', 'M', 'V'.

# Fonctionnement du programme

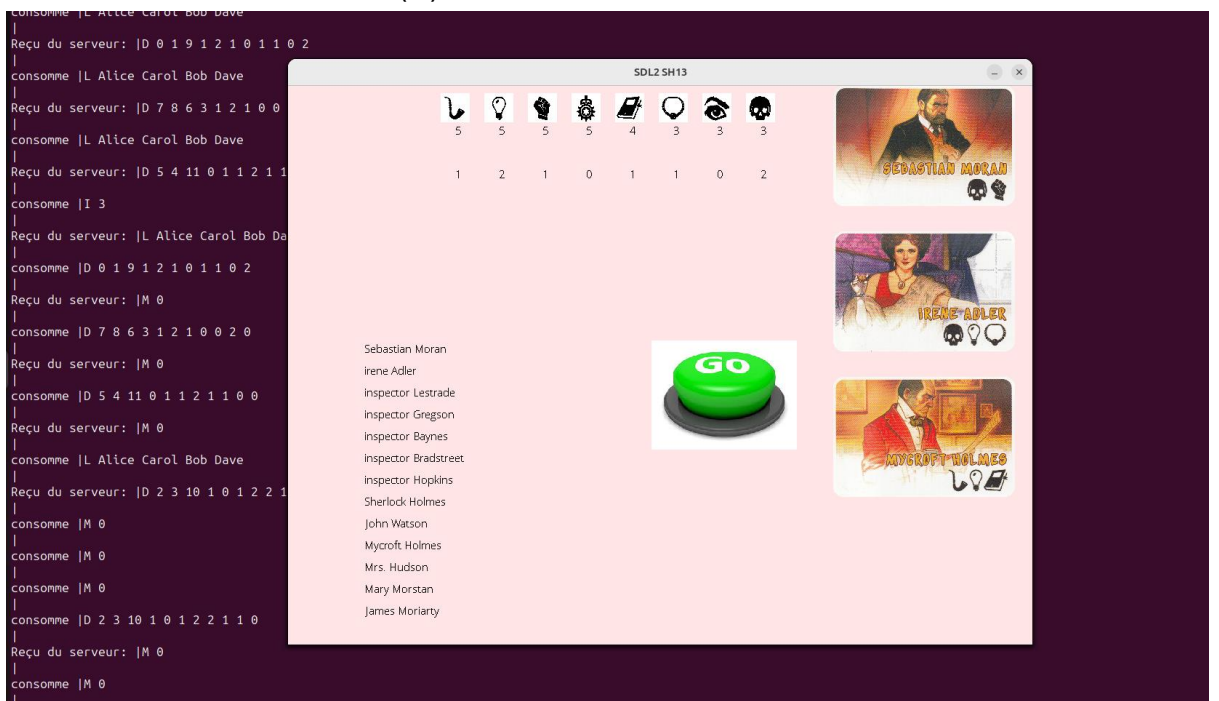
1. Lancement du serveur : './server 12345'

```
salma@salma-HP-Laptop-14s-fq1xxx:~/Documents/Systeme_exploit/PROJETS$ ./server 12345
0 Sebastian Moran
1 irene Adler
2 inspector Lestrade
3 inspector Gregson
4 inspector Baynes
5 inspector Bradstreet
6 inspector Hopkins
7 Sherlock Holmes
8 John Watson
9 Mycroft Holmes
10 Mrs. Hudson
11 Mary Morstan
12 James Moriarty
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
4 inspector Baynes
3 inspector Gregson
0 Sebastian Moran
5 inspector Bradstreet
10 Mrs. Hudson
8 John Watson
7 Sherlock Holmes
9 Mycroft Holmes
11 Mary Morstan
2 inspector Lestrade
6 inspector Hopkins
1 irene Adler
12 James Moriarty
00 01 02 02 01 00 00 01
02 00 02 01 00 01 01 00
02 02 01 00 02 01 00 00
01 01 00 02 01 01 02 01
```

2. Connexion de 4 clients, chacun avec un port propre.



3. Le serveur assigne 3 cartes par joueur + la carte du coupable (la 13ème).
4. Tour par tour, chaque joueur peut :
  - a. Interroger un autre joueur (S)
  - b. Interroger un objet (O)
  - c. Faire une accusation (G)



5. Si l'accusation est correcte, le joueur gagne. Sinon il est éliminé.

```
salma@salma-HP-Laptop-14s-fq1xxx: ~/Documents/Systeme_exploi/PROJ...
|
consomme |D 10 12 6 2 1 0 1 0 1 1 1
|
Reçu du serveur: |M 0
|
consomme |M 0
|
consomme |M 0
|
consomme |D 3 11 4 0 1 1 2 2 1 0 0
|
Reçu du serveur: |M 0
|
consomme |M 0
|
consomme |M 0
|
go! joueur=-1 objet=-1 guilt=9
Received packet from 127.0.0.1:41452
Data: [G 0 9
]
Reçu du serveur: |WIN Carol
```

6. Les informations sont transmises à tous via des messages texte.

## Concepts systèmes appliqués

### Sockets TCP

Utilisées pour les connexions entre serveur principal et les clients, mais aussi entre clients et leur propre serveur TCP interne. Fonctions utilisées :

- socket()
- bind()
- listen()
- accept()
- connect()
- write()
- read()

### Threads

Chaque client crée un thread pour écouter les messages du serveur en arrière-plan. Cela évite le blocage de la boucle SDL.

### Mutex

Synchronisation d'accès à 'gbuffer' via 'pthread\_mutex\_t'.

## Processus

Chaque client est lancé comme processus séparé via le script 'cmd.sh', ce qui simule des connexions réelles.

## Problèmes rencontrés et solutions

- **Erreur de port déjà utilisé** : le port 12345 était parfois bloqué. Solution : tuer l'ancien serveur avec 'kill -9'.
- **SDL.h manquant** : installation nécessaire des paquets :  
`sudo apt install libsdl2-dev libsdl2-image-dev libsdl2-ttf-dev`
- **Cartes toujours dans le même ordre** : oubli de 'srand(time(NULL))' pour mélanger.

## Conclusion

Ce projet a été une expérience enrichissante et très concrète pour mettre en pratique les notions vues en TP. Au début, j'ai trouvé le projet un peu compliqué car je ne connaissais pas le jeu Sherlock 13 et je ne comprenais pas comment il se jouait. Mais après avoir testé et observé plusieurs parties, tout est devenu plus clair.