Mai 2020

Projet Chathack
Manuel utilisateur

Network programming

AGULLO Vincent
CRETE Jonathan

Filière Informatique

Introduction

Ce manuel explique brièvement comment utiliser l'application ChatHack

1 - Lancer le serveur de base de données

Ouvrir un terminal et se placer dans le répertoire du projet.
Le serveur de base de donnée est un jar nommée **ServerMDP.jar**

```
>> java -jar ServerMDP.jar [Port d'écoute] [Fichier de mot de passe]
```

Par exemple :

```
>> java -jar ServerMDP.jar 5555 passwords.txt
```

Ce qui donne le résulat suivant dans la console :

```
jonathan@jonathan-ROG:/media/sf_GitHub/agullo-crete/Project_ChatHack$ java -jar ServerMDP.jar 5555 pass
ords.txt
mai 31, 2020 11:17:14 PM fr.upem.net.tcp.ServerMDP <init>
INFO: fr.upem.net.tcp.ServerMDP starts on port 5555with file passwords.txt
mai 31, 2020 11:17:14 PM fr.upem.net.tcp.ServerMDP launch
INFO: Server started
```

Si le message de log “Server started” s’affiche alors cela signifie que le serveur à bien demarré

2 - Lancer le serveur ChatHack

Se placer dans le répertoire **../jar** du projet.

```
>> java -jar ServerMDP.jar [Port d'écoute du client] [adresse d'hôte] [port du serveur de base données]
```

Par exemple :

```
>> java -jar ServerChatHack.jar localhost 5555
```

Ce qui donne le résultat suivant dans la console :

```
jonathan@jonathan-ROG:/media/sf_GitHub/agullo-crete/Project_ChatHack/jar$ java -jar ServerChatHack.jar
666 localhost 5555
The selector contains:
        Key for ServerSocketChannel : OP_ACCEPT
        Key for Client localhost/127.0.0.1:5555 : OP_READ
Starting select
```

Ceci implique que le serveur est bien démarré.

3 - Lancer le client ChatHack

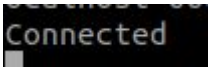
Se placer dans le répertoire **../jar** du projet.

```
>> java -jar ClientChatHack.jar [adresse hôte]  [port] [nom du fichier contenant les login/mot de passe]  [login]  [password] (optionnel)
```

Par exemple si vous souhaitez vous authentifer avec un mot de passe :

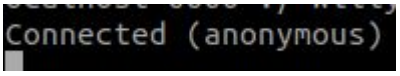
```
>> java -jar ClientChatHack.jar localhost 6666 ./ arnaud townhall
```

Ce qui produit, si le login et mot de passe correspondent :

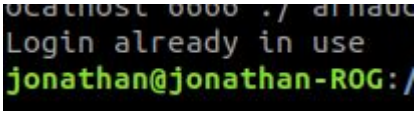


Ou de manière anonyme :

```
>> java -jar ClientChatHack.jar localhost 6666 ./ willy
```

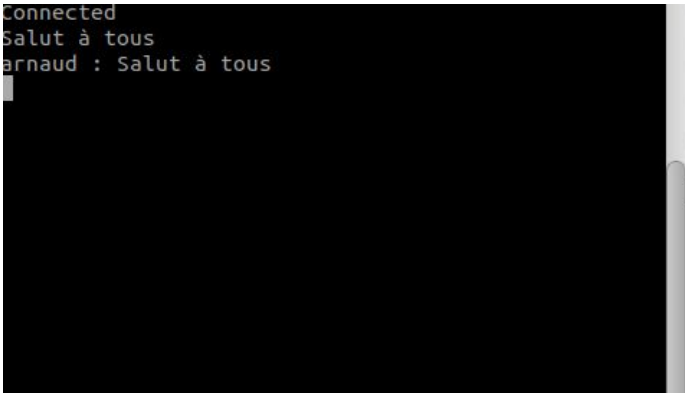


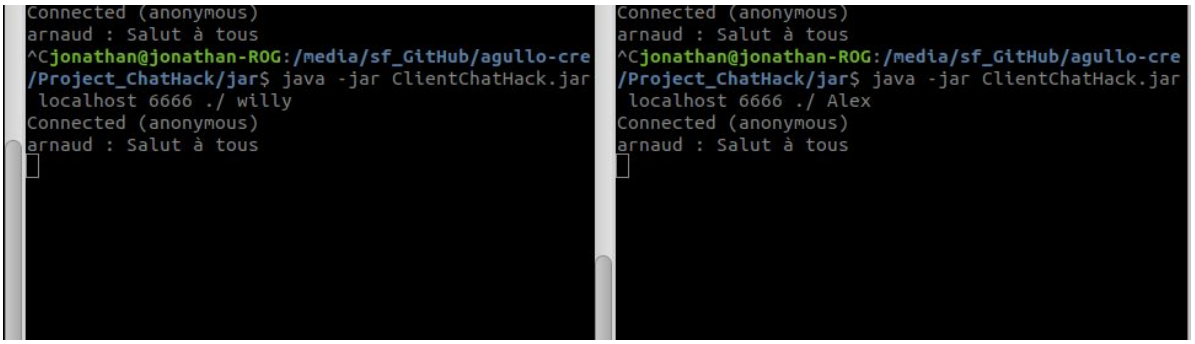
En revanche, si vous cherchez à vous authentifier avec un login déjà en cours d'utilisation vous aurez le message suivant, et vous ne pourrez pas vous connecter au serveur de chat :



4 - Envoyer un message public

Une fois connecté, pour envoyer un message public il suffit d'écrire directement dans la console :





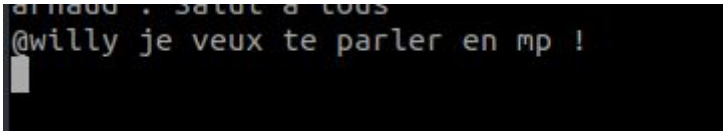
ici l'utilisateur arnaud à écrit directement les message “Salut à tous” dans la console et les utilisateurs willy et Alex ont bien reçu le message !

4 - Demande de connexion privée

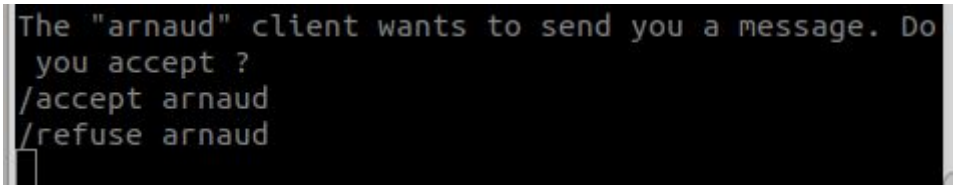
Imaginons que arnaud soihaite se connecter avec willy, il va alors taper la commande suivante :

```
>> @login message
```

```
>> @willy je veux te parler en mp !
```



L’utulisateur (client) willy peut alors accepter la demande de connexion avec la commande **/accept** [utilisateur] ou la refuser avec la commande **/refuse** [utilisateur]



S’il accepte la connexion est établie :

client Willy :

```
VERIFICATION --> OK
arnaud : je veux te parler en mp !
```

Client Arnaud :

```
0:0:0:0:0:0:0:0
The client has accept your request :) | Informations Login :willy InetSocketAddress /0:0:0:0:0:0:0:0 Con
nectId -4404704331130585409
```

la connexion est alors établie.
Au contraire, si Alex refuse la demande de connexion, l'utilisateur arnaud va recevoir le message suivant :

```
@willy je veux te parler en mp !
The client "willy" has refused your request !
```

4 - Envoi de message privé

Une fois la connexion établie les deux clients peuvent se parler entre eux avec la commande **@login message**

```
willy salut
illy : comment tu vas ?
```

```
arnaud : salut
@arnaud comment tu vas ?
```

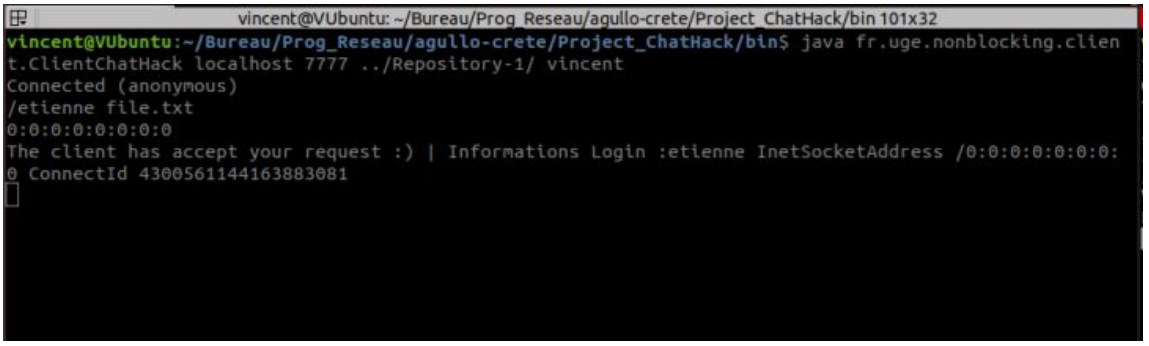
4 - Envoi de fichier privé

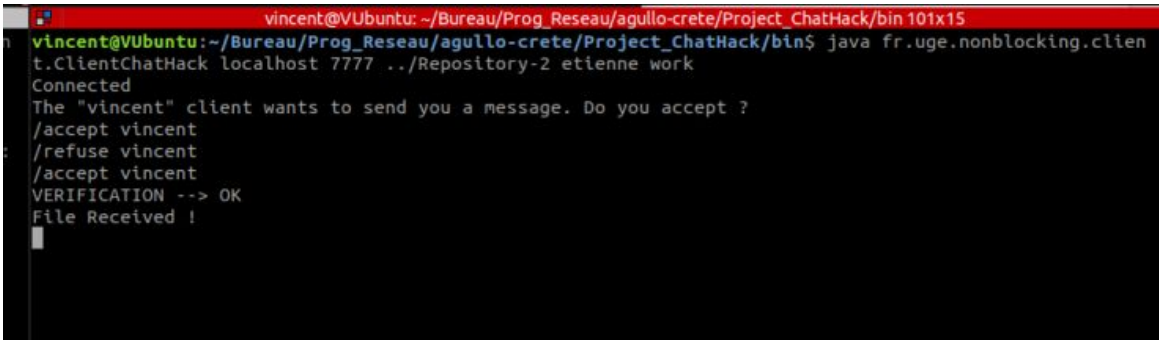
Les utilisateurs connecté peuvent s'envoyer des message avec la commande

>> /login + fichier

Par exemple :

>> /etienne file.txt





Avec cette application vous pouvez (Ce qui est fonctionnel)

- Authentification d'un utilisateur avec un pseudonyme et un mot de passe
    - Acceptation d'une demande de connexion privée
    - Refus d'une demande de connexion privée
- Authentification d'un utilisateur avec un pseudonyme uniquement
- Envoi de messages publiques entre tous les utilisateurs connectés
- Réception d'un message public
- Négociation d'une connexion privée
    - Acceptation d'une demande de connexion privée
    - Refus d'une demande de connexion privée
- Envoi de messages privés entre deux utilisateurs connectés
- Réception de messages privés entre deux utilisateurs connectés
- Envoi de fichiers privés entre deux utilisateurs connectés
- Réception de fichiers privés entre deux utilisateurs connectés

Ce qui ne fonctionne pas

Toutes les fonctionnalités de base d'un chat sont disponibles

Bugs connus

Aucun bug connu n'est à noter ici