MARTIN Jean Christophe

**Projet réalisé par :**

Projet ZABBIX

Sommaire

Table des matières

[I. Installation du Serveur 3](#_Toc154754669)

[1. Installation pour Windows 3](#_Toc154754670)

[2. Installation pour Debian/Ubuntu 4](#_Toc154754671)

[II. Installation du Client 6](#_Toc154754672)

[III. Documentation Serveur 7](#_Toc154754673)

[1. Classe 7](#_Toc154754674)

[2. Fonctions 8](#_Toc154754675)

[3. Commandes 12](#_Toc154754676)

[4. Base de données 13](#_Toc154754677)

[5. Droits d’accès 14](#_Toc154754678)

[IV. Documentation Client 15](#_Toc154754679)

[1. Interface Utilisateur 15](#_Toc154754680)

[2. Fonctions 18](#_Toc154754681)

[V. Document Réponse 19](#_Toc154754682)

[1. Les limites du programme 19](#_Toc154754683)

[2. La Sécurité 19](#_Toc154754684)

[3. Confidentialité 19](#_Toc154754685)

[4. Maintenance 19](#_Toc154754686)

# Installation du Serveur

## Installation pour Windows

1. **Installer Python pour Windows
2. Executez le fichier *client\_install.bat*
3. Installez MySQL serveur
4. Importer le fichier zabchat.sql sur le serveur MySQL
5. Lancez le script server-bdd.py

Optionnel : Pour aller plus loin, je vous propose d’utiliser phpMyAdmin afin de faciliter la modération du chat

## Installation pour Debian/Ubuntu

1. Installer Python

**Installation des dépendances**

sudo apt install build-essential zlib1g-dev libncurses5-dev libgdbm-dev libnss3-dev libssl-dev libreadline-dev libffi-dev libsqlite3-dev wget libbz2-dev

**Installation de Python**

sudo apt install python3 -y

sudo apt install python3-pip -y

1. Exécutez les commandes suivantes afin d’installer les modules

pip3 install mysql-connector-python

pip3 install cryptocode

pip3 install datetime

1. Installer Mysql

sudo apt update

sudo apt upgrade

sudo apt install <https://dev.mysql.com/get/mysql-apt-config_0.8.29-1_all.deb>

2 boites de dialogues vont s’ouvrir, sélectionnez OK

sudo apt update

sudo apt install mysql-server

sudo mysql\_secure\_installation

Entrez le nouveau mot de passe d’administration

En cas d’erreur voir la page : <https://www.nixcraft.com/t/mysql-failed-error-set-password-has-no-significance-for-user-root-localhost-as-the-authentication-method-used-doesnt-store-authentication-data-in-the-mysql-server-please-consider-using-alter-user/4233>

1. Clonez le répertoire GitHub du projet

gh repo clone JacquesCentue/TP-exeption

Ou copier les fichiers dans le répertoire de travail

1. Déployez la base de données d’administration

cd ./TP-exception/SAE/Server

mysql -u root -p < zabchat.sql

Note : il se peut que les paramètres de sécurité de mysql entraine une erreur dut au mot de passe de l’utilisateur « zabchat » crée afin d’administrer la base de données : <https://www.tecmint.com/fix-mysql-error-1819-hy000/>

1. Lancez le programme python

cd <répertoire ou se trouve le fichier server-bdd.py>

python3 server-bdd.py

1. Installation terminée

# Installation du Client

1. Installez Python 3
2. Copiez les fichiers clients
3. Exécutez le fichier client\_install.bat
4. Modifiez le fichier client-bdd-gui.py afin de modifier l’adresse IP du serveur

ATTENTION : il faut modifier l’IP a 2 endroits dans le code :

* Dans la classe ***AuthWindow***
* Dans la classe ***ChatWindow***

Sous l’attribut *self.HOST*

1. Le client est prêt à l’emploi

Pour aller plus loin vous pouvez aussi transformer le fichier python en exécutable via ce programme : <https://pypi.org/project/auto-py-to-exe/>

Comme cela, le programme n’est plus consultable par l’utilisateur

# Documentation Serveur

Le programme possède plusieurs modules permettant l’ajout de fonctionnalités :

* Socket : permet d’établir la communication entre le client distant et le serveur
* Threading : permet d’effectuer plusieurs fonctions en même temps
* mysql.connector : permet la connexion du serveur a la base de données pour administration
* cryptocode : permet de mettre en place un processus de chiffrement des messages entre le client et le serveur
* datetime : permet d’avoir la date et l’heure, utile lors du Timeout ou Kick
* hashlib : permet de hasher les mots de passe afin de ne pas les stocker tels quel dans la base de donnée et lors de l’authentification du root

## Classe

Le programme du serveur est composé de 1 classe qui comporte 10 attributs :

* *self.HOST* : adresse IP du serveur
* *self.PORT* : port de connexion cette valeur doit être identique aux clients
* *self.serverstatus* : statut du serveur
* *self.db\_connection* : ce sont les informations de connexion au serveur de base de données
* *self.clients* : liste de tous les clients connectés
* *self.ban* : liste des utilisateur/IP bannis
* *self.mdpRoot* : hash du mot de passe du root (par défaut : root)

Lors de l’appel du constructeur, les threads sont alors démarrés afin que le programme puisse faire plusieurs choses à la fois

## Fonctions

Dans le programme du serveur, il y a 6 fonctions existantes :

1. Le constructeur
2. listen\_connections

Cette fonction est démarrée dans un thread lors du démarrage du programme cela permet alors d’écouter si un nouveau client souhaite se connecter.

Si un client souhaite se connecter, alors la fonction va créer un thread qui s’occupera alors de gérer les clients en envoyant les arguments nécessaires tel que son adresse IP et son port.

1. handle\_client

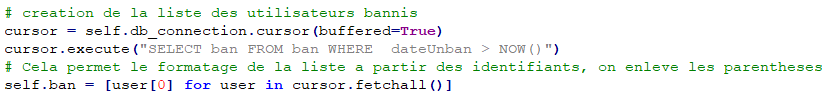
Cette fonction est appelée par le thread listen\_connections il va d’abord effectuer des tests si l’utilisateur existe ou si l’utilisateur n’est pas banni. Pour cela on va faire appel a la base de données qui renvoie toute sa liste des comptes et IP bannies puis on les stock dans *self.ban*

Figure 1 mise en place de la liste des utilisateurs bannis

ensuite on teste si les paramètres du client correspond a une entrée de la liste.

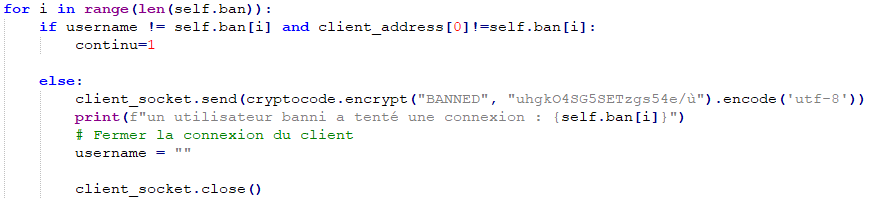


Figure 2 test si l'utilisateur est banni

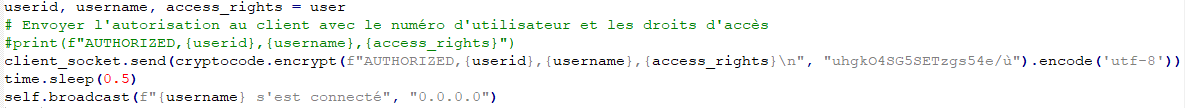
Si ce test revient positif, on va alors envoyer un message d’erreur au client et fermer sa connexion. Si il revient négatif, on peut continuer les test afin de déterminer ses droits d’accès afin d’afficher les canaux auquel l’utilisateur est autorisé a lire, pour cela, on va lui donner une séquence qui sera analysé par le client.

Figure 3 : Réponse du serveur au client avec les droits d'utilisateur

Après cela, le serveur se met en attente de message du client

Le serveur gère les canaux ce chat par le bais d’un préfixe avant chaque messages (/General, /Blabla,…) cela permet au serveur de bien identifier a quel canal le message est destiné et cela permet alors de sauvegarder le message dans la table de la base de donnée correspondante.

Après le message sauvegardé, le message est alors transmis a la fonction *broadcast.*

1. broadcast

Cette fonction va alors recevoir le message du client et le retransmettre à tous les clients connectés y compris de client destinataire, cela permet au destinataire de savoir si la connexion est établie.

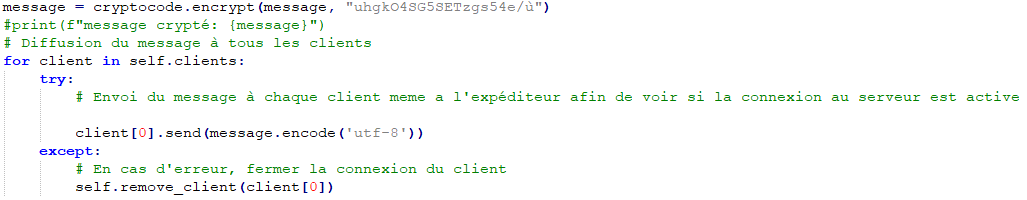


Figure 4 : Fonction Broadcast afin de diffuser les messages

1. remove\_client

Cette fonction permet de supprimer les clients de la liste des clients si une exception est levée par une autre fonction.

La fonction va parcourir la liste et supprimer l’index correspondant au client à supprimer

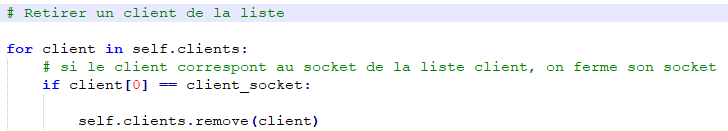


Figure 5 : Fonction remove\_client

1. commande

Cette fonction permet l’administration complète de l’application, cette fonction est gérée dans un thread, ce qui permet l’exécution des commandes en même temps que l’écoute des connexions et des messages.

Les commandes suivantes sont gérées :

- /help permet d'afficher la liste des commandes disponibles

- /ban <arg1> <arg2>(optionnel) permet de bannir un utilisateur et son adresse IP

- /unban <arg1> permet de gracier un utilisateur banni

- /send /<Canal>(optionnel) <message> permet d'envoyer un message via le serveur

- /kick <utilisateur> <jour(s)> permet de bannir temporairement un utilisateur

- /droit <numéro de droit> <utilisateur> permet de changer les droits de l'utilisateur -> droit se fait par rapport au numéro : voir readme

- /createuser <utilisateur> <mot de passe> <numéro de droit> permet de créer un utilisateur via la console

- /deluser <utilisateur> permet de supprimer un utilisateur via la console

- /mdp <utilisateur> <mot de passe> permet à l'administrateur de changer le mot de passe depuis la console")

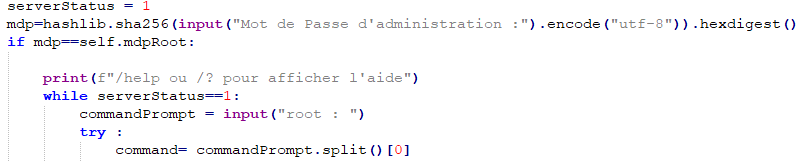
Lorsque l’administrateur envoie la commande, celle-ci est traitée par le serveur : le serveur va analyser le préfixe(ou premier mot) de la chaine de caractère et en fonction du résultat, la commande correspondante est exécutée.

Figure 6 : premières lignes de la fonction commandes

NB : *command= commandPrompt.split()[0]* permet de sauvegarder le préfixe de la commande afin d’effectuer les tests qui s’ensuivent.

## Commandes

## Base de données

## Droits d’accès

# Documentation Client

## Interface Utilisateur

1. Fenêtre d’authentification
2. Fenêtre principale
3. Fenêtre de changement de mot de passe

## Fonctions

# Document Réponse

## Les limites du programme

## La Sécurité

## Confidentialité

## Maintenance