Réseaux pour ingénieurs
GLO-2000

TP4 : Serveur de courriels

Professeur responsable :
Ronald Beaubrun
Ronald.Beaubrun@ift.ulaval.ca

Responsables des travaux pratiques :
Pierre Wendling et Vincent Primpied
GLO2000.laboratoires@gmail.com

Université Laval
Faculté des sciences et de génie
Automne 2022

Ce dernier TP a pour but de consolider vos acquis avec la programmation socket, avec la transmission de structures de données et la gestion simultanée de plusieurs clients.

Modalités de remise

- À faire en équipe de 3 (2 en cas d'un abandon).
- Les fichiers à remettre sont :
  - Le fichier `TP4_server.py`.
  - Le fichier `TP4_client.py`.
  - Les fichiers sont à rendre dans une unique archive zip avec l'organisation suivante :

    ```
    TP4_equipe_<votre numéro d'équipe>.zip
     ├── TP4_server.py
     ├── TP4_client.py
    ```

  - Notez qu'aucun des modules fourni n'est à inclure dans l'archive.

  N. B. : Une pénalité de 5% sera appliquée pour chaque point non respecté.
- Remise uniquement via le portail des cours.
- Date limite de remise est la suivante : **23 novembre 2022 à 23h00 (fuseau horaire de Québec)**.
- Tout travail remis en retard se verra attribuer la note **0**.
- Vous devez impérativement suivre les modèles `TP4_server.py` et `TP4_client.py` qui vous sont fournis.

Assurez-vous de respecter attentivement le protocole à utiliser pour échanger des messages. **L'envoi de messages dans un format différent fera échouer les tests et impactera grandement votre note. De même pour les modules à utiliser.**

Mise en situation

Vous avez été mandaté par une entreprise afin de leur concevoir un système de courriel interne (*@glo2000.ca*). Le système doit être composé d'un client et d'un serveur. Le serveur sera un relais SMTP qui traitera les envois de courriels à l'externe et à l'interne. Les utilisateurs utiliseront le client afin d'envoyer et de consulter leurs courriels.

Voici deux situations illustrant le fonctionnement du programme. Notez que votre implémentation n'a pas besoin de reproduire l'exemple au caractère près, cependant le même comportement est attendu.

La première situation couvre la création de comptes et l'envoi de courriels :

```
python TP4_client.py -d 127.0.0.1
Menu de connexion
```

```
1. Créer un compte
2. Se connecter
3. Quitter
Entrez votre choix [1-3]: 1
Entrez un nom d'utilisateur: Qing Long
Entrez un mot de passe:
La création a échouée:
 - Le nom d'utilisateur est invalide.
 - Le mot de passe n'est pas assez sûr.
Menu de connexion
1. Créer un compte
2. Se connecter
3. Quitter
Entrez votre choix [1-3]: 1
Entrez un nom d'utilisateur: Qing.Long
Entrez un mot de passe:
Menu principal
1. Consultation de courriels
2. Envoi de courriels
3. Statistiques
4. Se déconnecter
Entrez votre choix [1-4]: 2
Entrez l'adresse du destinataire: kohryu@glo2000.ca
Entrez le sujet: Bonjour!
Entrez le contenu du courriel, terminez la saisie avec un '.' seul sur une ligne:
Example de contenu
Sur plusieurs lignes
.
Menu principal
1. Consultation de courriels
2. Envoi de courriels
3. Statistiques
4. Se déconnecter
Entrez votre choix [1-4]: 4
Menu de connexion
1. Créer un compte
2. Se connecter
3. Quitter
Entrez votre choix [1-3]: 3
```

La seconde situation couvre la connexion à un compte, la consultation du courriel et des statistiques :

```
python TP4_client.py -d 127.0.0.1
Menu de connexion
1. Créer un compte
2. Se connecter
3. Quitter
Entrez votre choix [1-3]: 2
Entrez votre nom d'utilisateur: Kohryu
Entrez votre mot de passe:
```

```
Nom d'utilisateur ou mot de passe invalide.
Menu de connexion
1. Créer un compte
2. Se connecter
3. Quitter
Entrez votre choix [1-3]: 2
Entrez votre nom d'utilisateur: Kohryu
Entrez votre mot de passe:
Menu principal
1. Consultation de courriels
2. Envoi de courriels
3. Statistiques
4. Se déconnecter
Entrez votre choix [1-4]: 1
#1 Qing.Long@glo2000.ca - Bonjour! Tue, 12 Jul 2022 19:21:57 +0000
Entrez votre choix [1-1]: 1
De : Qing.Long@glo2000.ca
À : kohryu@glo2000.ca
Sujet : Bonjour!
Date : Tue, 12 Jul 2022 19:21:57 +0000
----------------------------------------
Example de contenu
Sur plusieurs lignes

Menu principal
1. Consultation de courriels
2. Envoi de courriels
3. Statistiques
4. Se déconnecter
Entrez votre choix [1-4]: 3
Nombre de messages : 1
Taille du dossier : 447 octets
Menu principal
1. Consultation de courriels
2. Envoi de courriels
3. Statistiques
4. Se déconnecter
Entrez votre choix [1-4]: 4
Menu de connexion
1. Créer un compte
2. Se connecter
3. Quitter
Entrez votre choix [1-3]: 3
```

Modules à utiliser

Vous aurez à utiliser plusieurs modules dans le cadre de ce travail pratique. Cette section vous présente ceux qui sont nouveaux.

Modules standards

Ces modules sont inclus avec les versions récentes de Python disponibles sur `www.python.org`. Assurez-vous d'utiliser une version récente de Python 3.

— Vous pouvez hacher une séquence d'octets avec les fonctions du module 'hashlib' et les comparer de manière sécurisée avec le module 'hmac' :

```python
import hashlib
import hmac

encoded_text = "Une chaine de caractère à hacher.".encode('utf-8')
expected_hash = "2bfdcfc01b9532dd071527441a62ef2983f7dba9253605a58c99d7c7"
hasher = hashlib.sha3_224()
hasher.update(encoded_text)
hmac.compare_digest(hasher.hexdigest(), expected_hash)
```

— Les modules 'pathlib' et 'os' permet de manipuler les chemins de fichiers indépendamment du système d'exploitation :

```python
import os
import pathlib

documents_path: pathlib.Path = pathlib.Path("Documents")
for file in documents_path.iterdir():
  file_size: int = os.path.getsize(file)
```

— Le module 'getpass' permet de récupérer un mot de passe sans l'afficher dans le terminal :

```python
import getpass

password = getpass.getpass("Entrez votre mot de passe : ")
```

— Vous pouvez également utiliser les modules présentés lors des séances précédentes tels que : email, json, logging, re, socket, select, …

Modules du TP4

Deux modules vous sont fournis et sont à utiliser pour la correction automatique :

— Le module 'glosocket' est le même que celui du TP3. Vous devez l'utiliser pour toutes vos communications entre le client et le serveur.

— Le module 'gloutils' vous fournit des constantes, gabarits, une énumération et des classes d'annotations :

```python
import gloutils

# Utilisation d'un gabarit
```

```
gloutils.EMAIL_DISPLAY.format(
    source="foo@glo2000.ca",
    destination="bar@glo2000.ca",
    subject="Exemple",
    date=gloutils.get_current_utc_time(),
    content="Corps du message"
)

# Création d'un message conforme au protocole
message = gloutils.GloMessage(
    header=gloutils.Headers.ERROR,
    payload=gloutils.ErrorPayload(
      error_message="Une erreur est survenue."
    )
)
```

— **Notez que les tests unitaires requièrent l'utilisation de ces modules**.

Spécifications des programmes

Le client et le serveur doivent impérativement :

— Utiliser le mode IPv4.
— Utiliser le mode TCP.
— Utiliser le module 'glosocket' pour échanger des messages.
— Utiliser les constantes/gabarits/énumérations/classes du module 'gloutils'.

Spécifications du client (8pts)

Constructeur de la classe : (1pt)

— Crée un socket et le connecte au serveur.
— Si la connexion est impossible, le constructeur fait appel à la méthode `sys.exit` avec un code différent de 0.

Tant que l'utilisateur n'est pas authentifié : (3pts)

— Le client affiche le menu de connexion à l'utilisateur.
— Si l'utilisateur souhaite créer un compte ou se connecter, le client lui demande un nom d'utilisateur avec `input` et un mot de passe avec `getpass`.
— Le client les transmet au serveur avec l'entête `AUTH_` appropriée.
— Si la réponse est `OK`, l'utilisateur est authentifié.
— Si la réponse est `ERROR`, le client affiche l'erreur et retourne au menu de connexion.
— Si l'utilisateur choisit de quitter, le client prévient le serveur avec l'entête `BYE` avant de fermer la connexion.

Une fois l'utilisateur authentifié, le client affiche le menu principal.

Consultation de courriel : (2pts)
- Le client demande la liste des courriels avec l'entête `INBOX_READING_REQUEST`.
- Si la liste contient au moins un courriel, elle est affichée, sinon le client retourne au menu principal.
- L'utilisateur choisit un courriel dans la liste.
- Le client transmet ce choix avec l'entête `INBOX_READING_CHOICE`.
- Le client affiche le courriel à l'aide du gabarit `EMAIL_DISPLAY` et retourne au menu principal.

Envoi de courriel : (1pt)
- Le client demande à l'utilisateur respectivement :
  - L'adresse de destination
  - Le sujet du courriel
  - Le contenu du courriel
- Le client récupère l'heure courante depuis le module 'gloutils'.
- Le client transfère les informations avec un entête `EMAIL_SENDING`.
- Le client affiche si l'envoi s'est effectué avec succès.

Consultation de statistiques : (1pt)
- Le client demande les statistiques du compte avec un entête `STATS_REQUEST`.
- Le client affiche les statistiques en utilisant le gabarit `STATS_DISPLAY`.

Déconnexion : (1pt)
- Le client informe le serveur de la déconnexion avec l'entête `AUTH_LOGOUT`.
- Le client retourne sur le menu de connexion.

Spécifications du serveur (12pts)

Constructeur de la classe : (2pts)
- Crée le socket serveur et le mets en écoute sur le port `APP_PORT`.
- Si l'opération échoue, le constructeur fait appel à la méthode `sys.exit` avec un code différent de 0.
- Prépare une liste vide pour les sockets clients connectés.
- Prépare un dictionnaire vide qui associe les sockets clients authentifiés à leur nom d'utilisateur.
- S'assurer que le dossier `SERVER_DATA_DIR` existe et qu'il contient le `SERVER_LOST_DIR`. Les créer sinon.

Une fois démarré le serveur attends sur `select.select()` que des clients se connectent/interagissent avec lui.

Connexion d'un client : (1pt)
- Le serveur accepte la connexion.
- Le serveur ajoute le client à la liste des sockets connectés.

Déconnexion d'un client : (1pt)
- Si nécessaire, le serveur retire le socket des utilisateurs connectés.
- Le serveur retire le socket de la liste des sockets connectés.

— Le serveur ferme le socket.

Création de comptes : (2pts)

— Le serveur s'assure que le nom d'utilisateur ne contient que des caractères alphanumériques, `_`, `.` ou `-`.
— Le serveur s'assure que le nom d'utilisateur n'est pas déjà pris, notez que les noms sont insensibles à la casse. Cela signifie que les noms 'BOB' et 'bob' correspondent au même utilisateur.
— Le serveur s'assure que le mot de passe est assez sécurisé. Le mot de passe doit avoir une taille supérieure ou égale à 10 caractères, contenir au moins un chiffre, une minuscule et une majuscule.
— Le serveur crée un dossier au nom de l'utilisateur dans le dossier `SERVER_DATA_DIR`.
— Le serveur hache le mot de passe de l'utilisateur avec l'algorithme `sha3_512` et l'écrit dans un fichier de son dossier nommé après la constante `PASSWORD_FILENAME`.
— Le serveur prévient le client du succès avec l'entête `OK`.
— Le serveur associe le socket du client à ce nom d'utilisateur.
— Si les identifiants sont invalides ou que le nom d'utilisateur est indisponible, le serveur répond avec l'entête `ERROR` et un message décrivant le problème.

Connexion à un compte : (1pt)

— Le serveur s'assure que le nom d'utilisateur existe.
— Le serveur hache le mot de passe et s'assure qu'il correspond à celui stocké dans le dossier de l'utilisateur.
— Le serveur prévient le client du succès avec l'entête `OK`.
— Le serveur associe le socket du client à ce nom d'utilisateur.
— Si les identifiants sont invalides, le serveur répond avec l'entête `ERROR` et un message l'accompagnant.

Consultation de la liste des courriels : (1pt)

— Le serveur récupère la liste des courriels depuis le dossier de l'utilisateur.
— Pour chaque courriel, le serveur récupère l'envoyeur, le sujet et la date.
— À l'aide du gabarit `SUBJECT_DISPLAY`, le serveur génère une liste de chaque sujet par ordre chronologique. La numérotation commence à 1 avec le courriel le plus récent.
— Le serveur transmet la liste au client avec l'entête `OK`.
— Si l'utilisateur n'a pas de courriel, le serveur transmet une liste vide.

Consultation d'un courriel : (0.5pt)

— Le serveur récupère le courriel associé au choix de l'utilisateur.
— Le serveur le transmet au client avec l'entête `OK`.

Consultation des statistiques : (1pt)

— Le serveur compte le nombre de courriels de l'utilisateur.
— Le serveur calcule le poids total du dossier de l'utilisateur.
— Le serveur transmet les données au client avec l'entête `OK`.

Envoi de courriel externe : (1pt)

— Le serveur crée un objet `email.message.EmailMessage`.
— Le serveur ajoute tous les champs reçus.
— Le serveur ouvre une connexion SMTP avec le serveur de l'université.
— Le serveur transfère le message au serveur SMTP.
— Le serveur indique au client si l'envoi s'est déroulé avec succès ou non avec respecti-

vement les entêtes `OK` et `ERROR`.

Envoi de courriel interne : (1pt)

- Le serveur vérifie que le destinataire existe.
- Le serveur utilise les méthodes du module 'json' pour écrire le payload tel quel dans le dossier du destinataire.
- Le serveur indique au client le succès de l'opération avec un entête `OK`.
- Si le destinataire n'existe pas, le serveur place le courriel dans le dossier spécial `SERVER_LOST_DIR` et répond au client avec un entête `ERROR` et un message d'erreur approprié.

Déconnexion d'un utilisateur : (0.5pt)

- Le serveur retire le socket du dictionnaire des utilisateurs connectés.

Environement de test

Les travaux seront testés dans une machine virtuelle basée sur l'image docker python:latest. Assurez-vous que votre travail fonctionne dans cet environnement, en particulier la manipulation de fichier.

Les tests utilisent la version de 'glosocket' et 'gloutils' qui vous ont été fournis. Vous ne pouvez donc pas modifier ces modules. Vous devez également utiliser impérativement les modèles TP4_server.py et TP4_client.py qui vous sont fournis.