TP socket : un serveur HTTP avec autorisation

1 Objectif

1. comprendre le fonctionnement global des Sockets et la notion de port de communication
2. appréhender l'API sockets en java en mode connecté et déconnecté (pour TCP et UDP)
3. écrire et utiliser un client UDP ;
4. écrire des clients et des serveurs en utilisant TCP et UDP
5. écrire une application web en utilisant le protocol HTTP (Hypertext Transfer Protocol)

2 Résumé du sujet

L'objectif final est de réaliser un serveur web qui demande l'autorisation.

Le navigateur se connecte (socket en mode connecté) au serveur http, envoie une requête http, attend une réponse http et l'affiche.

Le serveur http attend des demandes de connexions et crée une activité (un thread) pour chacune d'elles. Cette activité lit la requête, la décode, vérifie l'autorisation, la traite, et renvoie une réponse.

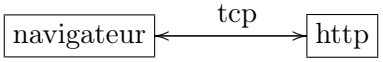
La requête envoyée par le navigateur est un texte de la forme :

```
GET /README HTTP/1.1
Host: posets.enseeiht.fr

```

Le mot `GET` identifie la méthode (obtention d'une page), et le chemin d'accès qui suit est la partie locale de l'URL, laquelle correspond, pour notre serveur, à un nom de fichier.

3 Étape 1 : Serveurhttp sans autorisation



— La seule requête HTTP acceptée est GET filename
— Le traitement consiste à renvoyer le contenu du fichier s'il existe, ou une erreur sinon.
— Les deux formats de réponse sont :

```
HTTP/1.1 200 Ok
Server: le mien
Content-Type: text/plain

contenu du fichier
```

ou

```
HTTP/1.1 400 Not Found
Server: le mien
Content-Type: text/plain

File not found
```

À faire : compléter `Serveurhttp.java` et tester :

puis dans un navigateur, charger l'url `http://localhost:8080/found` ou `http://localhost:8080/foo`.

4 Étape 2 : Le mécanisme d'autorisation

Le serveur Serveurhttpd doit maintenant avoir l'autorisation de répondre au navigateur. Pour cela, il diffuse un message en broadcast et attend une réponse. Si la réponse est négative ou s'il n'y a pas de réponse après un délai de garde (`time out`) :

```
HTTP/1.1 403 Forbidden
Server: le mien
Content-Type: text/plain

Forbidden access.
```

Noter que http est serveur du navigateur, et client du service d'authorisation : un processus n'est pas forcément uniquement client ou uniquement serveur.

À faire : compléter `ServeurAuthorisation.java` et mettre à jour `Serveurhttpd.java` en implémentant la méthode `byte[] check_authorisation()`.

Note : vu que le serveur d'autorisation est contacté en broadcast, il ne faut pas que tout le monde utilise le même numéro de port 9000. Le risque est en effet d'avoir plusieurs réponses pour une seule demande d'autorisation.

5 Étape 3 : Parallelisme

5.1 Multithreading

1. Tester l'application avec plusieurs clients. Commenter le résultat.
2. Modifier les serveurs implémentés dans les étapes précédentes pour pouvoir traiter plusieurs clients simultanément.

5.2 Thread pool

1. Quel est le nombre maximal de threads simultanés (qui pourrait être atteints) dans la version précédente ?
2. Modifier les serveurs implémentés dans les étapes précédentes pour pouvoir delimiter le nombre maximal de threads qui peuvent exécutés simultanément.