Programmation Réseaux
TP 2 – 1 ou 2 séance(s) de 2 heures

On souhaite implémenter un serveur web simple. Celui-ci ne renvoi que des pages web statiques. Créer un dossier TP_UE41 sur votre compte. Ce dernier contiendra l'ensemble des pages webs, photos, etc. de votre site. Dans un premier temps, votre site web ne gérera pas les photos ni les erreurs. Ces deux aspects pourront être fait au travers d'exercices complémentaires. Allez sur le site www.anthonybusson.fr, puis suivez « enseignement », « IUT ». Télécharger l'archive pageWeb.zip, et décompressez là dans le dossier TP_UE41 que vous avez créé. Votre code et vos exécutables devront se trouver dans ce dossier.

Fonctionnement du serveur :
Une fois qu'un client s'est connecté sur votre serveur (ouverture d'une connexion TCP), celui-ci enverra une requête HTTP en ASCII avec la syntaxe suivante (ici on demande la page web index.html sur l'adresse de loopback):

```
GET /index.html HTTP/1.1\r\n
Host: 127.0.0.1\r\n\r\n
```

Ou si on ne spécifie pas de page web, c'est la page par défaut qui est demandée:

```
GET / HTTP/1.1\r\n
Host: 127.0.0.1\r\n\r\n
```

La réponse de votre serveur va dépendre de plusieurs facteurs. Pour l'instant, nous allons considérer que le cas normal, sans erreur. Dans ce cas, l'en-tête HTTP de code 200 doit être renvoyé suivit de la page web demandée.

```
HTTP/1.1 200 OK\r\n
Content-Type: text/html\r\n\r\n
```
Ici la page web demandée…

Exercice 1 : Vers un serveur web

Il faut reprendre le code du dernier exercice de la feuille de TP1. Vous devrez le compléter. Dans cet exercice, vous devrez coder les tâches suivantes :

1. lors de la connexion d'un client, vous devrez récupérez le nom de fichier demandé par celui-ci ;
2. vous devrez renvoyer le code http ci-dessus (le code 200) indiquant que sa requête est correcte ;
3. vous devrez renvoyer le fichier demandé au client et fermer la connexion.

Dans un premier temps il faut lire les données envoyé par le client (déjà effectué dans l'exercice 4 du TP1 ; la chaîne de caractère hébergeant la requête devra être de 1000 octets). Pour la tâche 1, il faut utiliser la fonction se trouvant dans le fichier « parseRequest.txt ». Il se trouve dans l'archive que vous avez décompressé. Cette fonction prend comme argument la chaîne de caractère provenant du client, et renvoi dans l'argument `string` le nom du fichier demandé par le client. Cette fonction vérifie également que la syntaxe de la requête est

correcte, sinon la chaîne renvoyée est `NULL`. La fonction renvoi 0 en cas de succès, et -1 en cas d'erreur.

Pour la tâche 2, il suffit d'écrire/envoyer directement la chaîne de caractère donné plus haut sur la socket du client. Enfin, pour la tâche 3, il faut ouvrir le fichier demandé par le client, le lire et envoyer les données lues au client. A la fin du fichier, on ferme la socket du client.

Testez votre serveur avec votre navigateur en IPv4 et en IPv6. Il faut donner le nom d'un fichier qui existe dans votre répertoire, par exemple :

http://127.0.0.1:2000/index.html

et

http://[::1]:2000/index.html

en IPv6. Vous pourrez aussi tester sur les PCs d'un de vos voisins, il faudra alors indiquer l'adresse IP de votre PC.

Exercice 2 : fork()

Dans cet exercice, nous allons améliorer le code du serveur. La première étape consiste à boucler (`while`) sur l'instruction `accept()` afin que votre serveur puisse gérer de nouveaux clients. La seconde amélioration consiste à utiliser un autre processus pour gérer chaque client. La gestion d'un client devra se faire par une fonction plutôt que dans le `main`. Cela permettra d'aérer le code. Après l'appel à `accept()`, vous devez créer un nouveau processus avec l'appel à `fork()`. Le processus fils devra gérer la nouvelle connexion (traiter la requête, envoyer le fichier, etc.). Le processus père devra reboucler sur l'appel à `accept()` pour se remettre à l'écoute des connexions entrantes.

Note : n'oubliez, pas au niveau du père, de fermer la nouvelle connexion avant de reboucler.

Exercice 3 : Gestion des erreurs (facultatif)

Dans le cas ou une erreur se produit, le serveur renvoi un code spécial (autre que 200) et des pages web spéciales au client. Nous distinguerons deux types d'erreur, les erreurs du au fait que la syntaxe de la requête du client est invalide, et du au fait que le fichier demandé n'existe pas. Dans le premier cas la fonction `parseRequest()` renvoi -1, le code http et la page web (dans l'archive siteWeb.zip) qui doivent être renvoyé au client sont :

```
HTTP/1.1 400 BAD REQUEST\r\n
Content-Type: text/html\r\n\r\n
```
Ici la page web file400.html

Dans le deuxième cas l'appel à la fonction dont vous vous servez pour ouvrir le fichier (open, fopen, etc.) aura renvoyé un code d'erreur. Le code http et la page web (dans l'archive siteWeb.zip) qui doivent être renvoyé au client sont :

```
HTTP/1.1 404 FILE NOT FOUND\r\n
Content-Type: text/html\r\n\r\n
```

Ici la page web file404.html

Note : Une gestion encore plus fine des erreurs est possible. Si vous utilisez open pour ouvrir le fichier et qu'une erreur se produit, vous pouvez regarder la valeur de `errno`. Celle-ci vous permettra de connaître précisément l'erreur qui s'est produite (fichier non trouvé, problème de droits, etc.). En principe pour chaque type d'erreur, il y aura un code http différent (un problème de droit sera un code 500 par exemple : `internal server error`).

Exercice 4 : Gestion des images (facultatif)

Pour l'instant, votre serveur ne prend en charge que les fichiers textes (html et txt). Cet exercice va permettre de gérer également les images. Lors de la réponse du serveur au client, celui-ci indique le type de fichiers renvoyé (`Content-Type: text/html\r\n\r\n`). Cela permet au navigateur de savoir comment interpréter ce fichier : il ne traite pas une image de la même manière qu'un fichier ASCII. Lorsque le document demandé n'est pas un fichier texte, il faut simplement renvoyé le bon « `Content-Type` ». Pour une image ce sera :

```
HTTP/1.1 200 OK\r\n
Content-Type: image/png\r\n\r\n
```
Puis la photo…

Si c'est une image jpeg plutôt que png, vous devez remplacer png par jpeg dans le champ Content-Type. Dans cet exercice il vous est demandé de regarder le type d'extension du document demandé par le client. Si il s'agit d'une photo, modifier votre code pour qu'il renvoi le code approprié. Vous pourrez tester avec votre navigateur et une image se trouvant dans le dossier de votre site web.

Exercice 5 : Gestion des logs (facultatif)

Lorsqu'un client se connecte, vous pouvez récupérer son adresse IP et l'URL demandé (c'est ce que font tous les serveurs). Pour ce faire, il faut indiquer dans `accept()` un pointeur sur une structure sockaddr_in6 qui sera mis à jour par le système avec l'adresse IP du client.