



Systeme Unix

my_ftp

Contact

b-psu-330@epitech.eu



Table des matières

Détails administratifs	2
Sujet	3
.1 Serveur	3
.2 Client	3
.3 Bonus	4
.4 Super Bonus	4
Fonctions interdites	5
Fonctions autorisées	6

Détails administratifs

- Les sources doivent être rendues dans le répertoire PSU_année_myftp
ex : PSU_2013_myftp pour l'année 2013-2014
- L'intégralité de votre projet devra compiler avec un unique Makefile.
- Ce projet est à réaliser seul

Sujet

- Il s'agit dans ce projet de réaliser un client et un serveur FTP.
- Vous devez implémenter le protocole fourni en annexe du sujet (vos bonus devront respecter la RFC959).
- La communication réseau se fera au travers de sockets TCP.

.1 Serveur

- Synopsis :

```
1 Usage : ./serveur port
```

- `port` correspond au numéro de port d'écoute de la socket serveur.
- Le serveur doit supporter plusieurs client simultanément par l'intermédiaire d'un **fork**.



Indices

Si vous êtes très à l'aise avec **select**, vous pouvez l'utiliser. Néanmoins, vous aurez l'occasion de l'utiliser au cours du projet `my_irc`, profitez donc de ce mini-projet pour vous familiariser avec un serveur qui **fork**.



Si vous utilisez **select**, faites les choses correctement :

- **select** en lecture
- **select** en écriture
- serveur non bloquant

Si votre serveur est mal conçu parce que vous avez voulu utiliser **select**, vous perdrez de nombreux points en soutenance.

- Le serveur doit répondre aux exigences suivantes :
 - Une authentification avec un compte "Anonymous" sans mot de passe

.2 Client

- Synopsis :

```
1 Usage : ./client machine port
```

- `machine` correspond au nom (ou à l'adresse IP) de la machine sur laquelle se trouve le serveur.
 - `port` correspond au numéro du port d'écoute du serveur.
- Le client doit gérer les commandes suivantes :
 - `user` : spécifie l'utilisateur au serveur et demande un mot de passe si nécessaire.
 - `ls` : liste le répertoire courant du serveur.
 - `cd` : change le répertoire courant du serveur.

- get `__FILE__` : transfère le fichier `__FILE__` du serveur vers le client.
- put `__FILE__` : transfère le fichier `__FILE__` du client vers le serveur.
- pwd : affiche le répertoire courant du serveur.
- quit : coupe la connexion et sort du programme.
- Le client doit répondre aux exigences suivantes :
 - Un prompt spécifique au client (pour le différencier du shell)
 - Impossibilité de parcourir le "file system" à un niveau inférieur que le répertoire d'exécution du serveur (document root par défaut).
 - Affichage des codes retours (avec message explicatif) à chaque requête.

.3 Bonus

- lcd + lpwd + lls
- mget + mput
- gestion des droits
- home directory différent pour chaque compte utilisateur
- gestion des modes "bin" est "asc"
- complétion lors d'un get
- complétion lors d'un put

.4 Super Bonus

- respect total de la RFC

Fonctions interdites

- recv
- send
- tous les appels permettant de passer la socket en non bloquant

Fonctions autorisées

- la libC amputées des fonctions interdites