

**Un système de partage de fichiers en P2P**  
**Projet réseaux (Module Programmation IP)**  
**UE : Ingénierie des réseaux Tout IP**

## **Introduction**

La forme commune de stockage des données sont les fichiers, ce qui fait des protocoles de transfert de fichiers un moyen de base pour le partage et l'extraction des informations. Dans ce projet, vous devez concevoir un système de partage de données entre plusieurs utilisateurs basé sur un protocole de transfert de fichiers. Il doit permettre à ces utilisateurs d'accéder à un système distribué de stockage de données à partir d'une simple recherche par mots-clés.

Le choix du mécanisme de concurrence (fork, threads, etc) et des protocoles au niveau transport (TCP/UDP) vous est laissé. Votre système aura un principe similaire au service fournie par des logiciels *Peer to Peer* (P2P) du type Snapster, Kazaa, Grokster et Morpheus.

## **Implémentation**

Le projet est constitué de deux composantes :

1. Un client et un serveur de transfert de fichiers qui s'exécutent sur chaque site utilisateur. Le client permet à l'utilisateur d'accéder aux fichiers stockés sur des sites utilisateurs distants. Le serveur est responsable des services de transfert de fichiers demandés par un client distant.
2. un serveur centralisé qui :
  - a. permet aux utilisateurs d'accéder et de spécifier la description des fichiers partagés de leurs sites. Le mécanisme d'accès et utilisé pour authentifier un utilisateur.
  - b. fournit une recherche par utilisation de mots-clés. Le résultat d'une telle recherche est la localisation des fichiers qui répondent à la requête utilisateur.

La figure.1 représente l'architecture du système. Chaque hôte utilisateur dans le système a deux composantes, un client de transfert de fichiers et un serveur de transfert de fichiers. Quand le serveur de transfert de fichiers est lancé, il demande à l'utilisateur connecté son '*username*' et son '*password*' et envoie ces informations au serveur centralisé. Sur ce dernier aussi, vous avez besoin d'implémenter un mécanisme d'enregistrement des nouveaux utilisateurs. L'authentification peut être effectuée en utilisant le même mécanisme de *login/password* d'UNIX (Vous pouvez consulter la page man pour la fonction `crypt()`). Lors de l'authentification d'un utilisateur, il ne faut **jamais** envoyer un mot de passe utilisateur non crypté (en clair) sur le réseau.

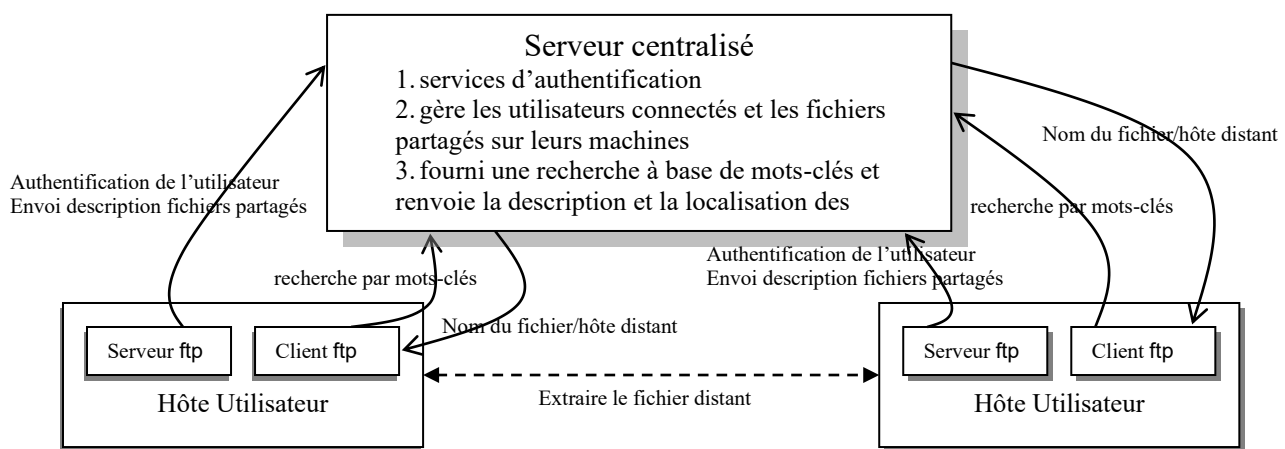
Le serveur centralisé vérifie l'identité de l'utilisateur et envoie un message de succès/échec basé sur le résultat de l'authentification. Si l'utilisateur a été authentifié avec succès, le serveur ftp du site utilisateur se charge de télécharger (a) le *hostname* du site utilisateur (b) les noms des fichiers partagés et (c) la description textuelle des fichiers partagés. Cette description textuelle est utilisée pour les recherches à base de mots-clés pour localiser un fichier partagé distant. Vous pouvez supposer que les fichiers partagés au niveau du site utilisateur se trouvent dans le même répertoire, bien qu'une implémentation améliorée permettra à un utilisateur de partager des fichiers se trouvant dans le même répertoire ou dans des sous répertoires différents. Le répertoire de fichiers partagés contient aussi la **description**

**des fichiers**, qui contient le nom et la description textuelle du contenu de chacun de ces fichiers. Par exemple, si vous avez un fichier partagé appelé MP3.txt, votre description peut être comme suit :

```
MP3.txt
FAQ sur l'algorithme de compression utilisé dans le codage audio MPEG-2
```

Vous pouvez utiliser n'importe quel format de fichier pour la description du fichier. Vous pouvez aussi rajouter d'autres champs comme *l'auteur du fichier*, *la date de création*, *la longueur du fichier* etc.

Vous aurez besoin de faire passer le *hostname* et le numéro de port du serveur central au client ftp aussi bien qu'au serveur ftp au niveau de la machine client soit par ligne de commande, soit en supportant une commande de connexion.



**Figure.1 : architecture du système de partage de fichiers en P2P**

Après que le serveur ftp démarre et télécharge les descriptions des fichiers partagés, les clients ftp peuvent maintenant envoyer leurs requêtes vers le serveur centralisé concernant la disponibilité des fichiers partagés. Pour ceci, le client ftp demande à l'utilisateur un mot-clé et contacte le serveur centralisé en lui remettant ce mot-clé. Le serveur centralisé accomplit une simple recherche par mot-clé sur les différentes descriptions des fichiers partagés et retourne la localisation des fichiers recherchés. Ce résultat contient les noms des hôtes qui contiennent les fichiers concernés et les noms de ces fichiers. Vous pouvez ne générer des résultats positifs à une recherche que si le mot clé entier existe dans des descriptions, mais une implémentation plus optimisée permettra de prendre le mot clé même s'il est une sous-chaîne de caractère dans une description de fichier partagé.

Après que le serveur centralisé retourne la localisation et le nom du fichier distant, le client ftp distant peut contacter le serveur ftp distant pour extraire le fichier. Le client ftp ne peut pas faire du 'upload', mais seulement du 'download'.

Vos clients et serveurs doivent inclure des mécanismes pour trier les résultats des recherches par mots-clés.

Le serveur centralisé doit être en mesure de (a) authentifier les utilisateurs et (b) effectuer des recherches par mots-clés sur les descriptions textuelles que les utilisateurs lui ont transféré auparavant. Pour l'authentification des utilisateurs, le serveur centralisé maintient une liste

des noms des utilisateurs enregistrés associés à leurs mots de passe cryptés. Quand un utilisateur s'enregistre pour la première fois, il doit spécifier son no d'utilisateur et son mot de passe.

Il faut prendre, aussi, en considération le cas où un serveur distant puisse se déconnecter. Dans ce cas, le serveur centralisé ne doit plus prendre en considération les descriptions des fichiers associés à ce serveur distant dans ces recherches. Pour simplifier la gestion de cette procédure, on considérera qu'un serveur distant ne peut se déconnecter qu'après avoir averti le serveur centralisé. Une façon améliorée de gérer cela sera de considérer que les serveurs distants contactent périodiques le serveur centralisé pour indiquer qu'ils sont toujours actifs, une fois cette notification n'est plus reçu au bout d'un certain temps, le serveur centralisé déduit qu'il y a eu déconnection.

Il faut aussi prendre en considération les points suivants :

1. plusieurs serveurs distants peuvent s'authentifier en même temps sur le serveur centralisé
2. quand un client télécharge un nouveau fichier partagé il doit l'inclure dans la liste de ses fichiers à partager avec les autres utilisateurs, et donc effectuer une mise à jour de sa liste auprès du serveur centralisé.

### **Evaluation et modalités de remise du projet**

Le projet est un travail à faire par groupe de 2 étudiants. On utilisera le langage C ou un autre langage.

*Date limite de retour : mardi 06 janvier 2026*