

Résumé

Les exercices précédents vous ont permis de développer des applications clients et serveurs en utilisant des protocoles de communication d'une grande simplicité.

Le but de ce TP est la mise en œuvre d'un protocole de communication plus complexe par dessus la couche de transport (TCP ou UDP). On espère ainsi vous faire comprendre l'importance de bien définir le protocole.

1 Introduction

Un protocole de communication est une suite de règles de communication qui permettent, lorsqu'elles sont suivies correctement, un dialogue clair entre une application cliente cherchant un service particulier, et une application serveur rendant ce service.

L'avantage de définir correctement de manière ouverte (open source) les protocoles de communication est de permettre à tous d'implémenter une application client ou serveur (ou les deux) permettant de réaliser le dialogue sans devoir se restreindre à une application spécifique.

L'ensemble des applications clientes respectant les règles du dialogue défini pour un protocole de communication donné doivent donc pouvoir fonctionner avec l'ensemble des applications serveurs respectant ce même protocole.

Un protocole de communication nécessite donc de définir :

1. L'**ordre** des messages constituant le dialogue ;
2. La **structure** des messages qui seront transmis entre les protagonistes (le client et le serveur) ;
3. Le **contenu** de ces messages ;
4. Définir tous les cas de figure possibles pouvant survenir lors d'un dialogue.

La structure du message transporté doit pouvoir convenir pour toutes les étapes du dialogue entre le client et le serveur. Et leur contenu doit être clairement identifié (et listé) pour éviter les imprévus.

Le protocole doit donc aussi prévoir les cas d'erreur et indiquer comment réagir dans tous les cas de figures.

 **Attention** : Le protocole ne décide pas de l'implémentation. Chacun est libre d'implémenter les règles du dialogue comme il l'entend. Mais les règles à respecter peuvent parfois contenir des impératifs techniques comme par exemple : conserver des informations prêtes à l'envoi pour envoyer un message complet uniquement à certains moments si cela est imposé par le protocole.

Nous allons dans la suite décrire brièvement le fonctionnement du protocole FTP.

2 File Transfert Protocole

La base principale du protocole FTP est décrite dans le [RFC 959](#)

 Récupérez ce fichier, ouvrez-le et parcourez-le en parallèle des informations qui suivent.

Le protocole FTP est un protocole de transferts de fichiers qui fonctionne au-dessus de TCP.

Il utilise deux canaux de communication entre le client et le serveur, et donc deux ports :

- Le canal de contrôle, sur lequel sont transmises les commandes du protocole, par défaut sur le port 21 ;
- Le canal de données, sur lequel sont transmises les données elles-mêmes, par défaut sur le port précédent le port du canal de contrôle (donc ici le port 20) ;

Il permet d'échanger des fichiers de façon très simple, selon cette séquence d'actions :

1. Ouverture de connexion avec login et mot de passe
2. Transfert de fichiers
3. Fermeture de connexion

Certains serveurs FTP autorisent les utilisateurs anonymes. Pour cela, un nom d'utilisateur particulier est utilisé : *anonymous*. N'importe quelle chaîne de caractères est alors acceptée comme mot de passe.

2.1 Architecture

Le client et le serveurs sont constitués de deux modules :

- Le *Protocol Interpreter* (PI), qui interprète les commandes reçues du client sur le canal de contrôle ;
- Le *Data Transfer Process* (DTP), qui effectue les échanges de données eux-mêmes (envois et réceptions) sur le canal de données.

Le PI commande le DTP : quand le client envoie une commande d'échange de données sur le canal de contrôle, le PI appelle le DTP pour lui demander de participer à cet échange de données.

Le client a également, en outre, une interface d'interaction avec le client, qui peut être graphique ou en ligne de commandes.

Dans un fonctionnement client-serveur, l'architecture utilisée par le protocole FTP peut-être représenté comme sur la figure 1. Chaque machine, cliente comme serveur, dispose de son propre système de fichier. Les fichiers sont transmis entre le client et le serveur (dans un sens ou dans l'autre) par les DTP sur le canal de données, sous le contrôle du PI.

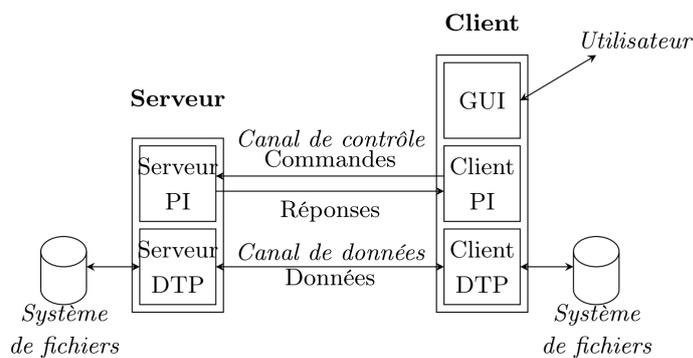


FIGURE 1 – Base du fonctionnement du protocole FTP

2.2 Protocole

Le protocole FTP fonctionne en mode texte. Les commandes et les réponses échangées entre les PI sont envoyées avec l'encodage ASCII 7 bits. Les communications sur le canal de données peuvent se faire en binaire ou en ASCII, selon le paramétrage demandé par le PI (commande TYPE).

Les commandes FTP sont constituées de trois ou quatre lettres majuscules.

On distingue trois types de commandes FTP

2.2.1 Les commandes de contrôle d'accès

Le contrôle d'accès définit les privilèges utilisateur nécessaires pour utiliser un système, et pour accéder à des fichiers dans ce système. Le contrôle d'accès est nécessaire pour éviter un usage accidentel ou non autorisé de ressources fichiers.

- USER : Identification de l'utilisateur
- PASS (password) : Mot de passe de l'utilisateur (suit immédiatement une commande USER)
- PWD (print working directory) : Affiche le chemin courant sur le FTP ;
- CWD (change working directory) : Changement de répertoire dans l'arborescence distante
- CDUP (change directory up) : Remonter dans l'arborescence distante
- QUIT : Fin de la session

Certaines commandes peuvent être

2.2.2 Les commandes du paramétrage de transfert

Configure le transfert. À n'utiliser que pour changer un paramètre par défaut non standard. Ces commandes sont donc à utiliser AVANT une commande de service.

- TYPE : Type de format d'échange des données (A=ascii, I=image)
- STRU (structure) : Structures échangées (F=file)
- MODE : Mode de transfert des données (S=stream)
- PORT : IP et numéro de port auquel le DTP-serveur doit se connecter
- PASV : Changer le IP/port de transfert des données en mode passif.

2.2.3 Les commandes de service FTP

Les commandes de service FTP rassemblent toutes les commandes opérationnelles de transfert ou système qui peuvent être invoquées par l'utilisateur. L'argument d'une commande de service est en général un chemin d'accès.

Les fichiers transmis suite à une commande de service, le sont sur le canal de données.

Les réponses à caractère informatif sont transmises sur le canal de contrôle.

Voici quelques exemples de commandes de services :

- RETR (retrieve) : Copie sur la machine locale
- STOR (store) : Copie sur la machine distante
- LIST : Liste des fichiers dans un répertoire ou information sur un fichier
- NLST : Liste des fichiers dans un répertoire
- MKD / RMD : Créer / supprimer un répertoire
- DELE : Effacer un fichier
- PWD : Nom du répertoire courant
- HELP : Information générale ou sur une commande
- NOOP (no operation) : aucune action (évite la déconnexion).

Vous retrouverez la liste des commandes de services dans le [RFC 959](#) en section 4.1 (explication des commandes) et 5.3 (format des commande et arguments).

2.3 Exemple

Voici un exemple typique d'échange entre un client (ici telnet) et un serveur.

Ne sont représentés ici que les messages texte du protocole FTP (pas les segments TCP et encore moins les datagrammes IP contenant ces messages).

```
1 login@host:~$ telnet ftp.free.fr 21
2 Trying 212.27.60.27...
3 Connected to ftp.proxad.net.
4 Escape character is '^]'.
5 220 Welcome to ProXad FTP server
6 USER anonymous
7 331 Please specify the password.
8 PASS toto
9 230 Login successful.
10 LIST
11 425 Use PORT or PASV first.
12 QUIT
13 221 Goodbye.
14 Connection closed by foreign host.
```

Comme on peut le voir, la commande LIST n'a pas donné le résultat escompté. En effet, la commande LIST nécessite l'utilisation d'un canal de données pour la réponse. Ce dernier pouvant être utilisé dans les deux directions, il existe deux possibilités de fonctionnement :

1. Le mode **actif** : le client (en utilisant la commande PORT) détermine le port d'écoute et joue le rôle de serveur pour ce canal ;
2. Le mode **passif** : le client (en utilisant la commande PASV) choisit le mode passif et c'est alors le serveur qui détermine le port d'écoute et joue le rôle de serveur pour ce canal.

Canal de transfert des données

1. Le mode actif est utilisé de la manière suivante :
 - Le client FTP envoie sa commande "PORT IP1,IP2,IP3,IP4,PORT1,PORT2 "
 - Le serveur répond qu'il est d'accord
 - Le client envoie sa commande (LIST, RETR, ...)
 - Le serveur se connecte sur le client sur le port spécifié par celui ci.
 - Le transfert se déroule de la même manière qu'en mode passif.
2. Le mode passif est utilisé de la manière suivante : Le client FTP envoie sur le canal de contrôle la commande PASV
 - Le Serveur répond "227 Entering Passive Mode (IP1,IP2,IP3,IP4,PORT1,PORT2)"
 - Le client décode la phrase, calcule le N° de Port = PORT1 * 256 + PORT2 (chaque numéro étant sur 1 octet)
 - Le client se connecte à l'adresse IP donnée (attention ce sont des virgules qui sépare les octets), sur le N° de Port donné et ouvre le port de transfert
 - Une fois la connexion ouverte, le client envoie sa commande sur le port de contrôle (LIST, RETR, ...)
 - Les données sont reçues sur le port data, et dès l'envoi terminé, le serveur envoie sur le port contrôle une phrase de fin de transfert, et ferme sa connexion de port de transfert. Attention le message de fin sur le port contrôle arrive souvent avant la fin de la réception des données sur le port transfert.

Le protocole FTP fonctionnant en mode requête/réponse, chaque demande est suivi d'une réponse. Elle est constituée d'un code à 3 chiffres suivi (optionnel) d'une explication courte. Le premier chiffre indique si la commande se termine en succès, échec, ou est incomplète. Les second et troisième chiffres permettent de détailler de plus en plus finement l'information apportée par le premier chiffre.

On se référera au RFC 959 section 4.2 pour plus de détail sur la signification des codes de réponse.

3 Implémentation

Pour réaliser ce TP, vous formerez des binômes.

Les ports 20 et 21 vous étant inaccessibles, vous utiliserez sur le serveur le **port 2121** comme port pour le canal de contrôle. Et donc, 2120 sera le port par défaut pour le canal de données.

Vous devrez réaliser une implémentation minimale comme décrit section 5.1 du rfc959 avec les ajouts/-modifications suivantes :

- Pas de commande STRU : Nous considérerons que la seule structure que nous utiliserons est un système de fichier standard
- MODE/TYPE : gestion **binaire** uniquement (mode S / type I)
- Un ensemble de commandes obligatoires sont indiqués dans les transparents de cours (cf Moodle, section du projet). Notamment les commandes d'authentification (USER, PASS), de gestion du canal de données (PORT, PASV), de gestion des fichiers (LIST, PWD, CWD) et bien sûr, les commandes de téléchargement ou de dépôt de fichier (RETR, STOR) ainsi que la commande HELP sont indispensables.

Cette liste **n'est pas** exhaustive.

Note

Vous pouvez bien sûr vous inspirer de ce que vous trouverez sur le net. Mais au final, ce doit être votre code que nous regarderons.

En plus de la RFC qui fait foi, voici un lien contenant des informations synthétiques et claires sur le protocole FTP : <http://cr.yp.to/ftp.html>