



**Installation d'un serveur FTP (File Transfert Protocol) sous
Ubuntu Server 12.10**



Table des matières



Installation d'un serveur FTP (File Transfert Protocol) sous Ubuntu Server 12.10	1
Transfert de fichiers, le protocole FTP (File Transfer Protocol).....	3
Généralités.....	3
Utilisation du mode FTP	7
Installation du serveur FTP et configuration.....	9
1. Topologie Physique et logique du réseau.....	9
2. Installer et configurer vsftpd	11
3. Configuration du serveur FTP.....	13
4. Utilisation du FTP avec Utilisateurs locaux.....	18
Répertoire de partage et Utilisateurs	18
Exploration des Trames avec le logiciel Wireshark.....	25
Configuration avec Utilisateurs virtuels.....	27
Création de la base de données	27

Transfert de fichiers, le protocole FTP (File Transfer Protocol)

Généralités

Le *File Transfer Protocol* (protocole de transfert de fichiers), ou FTP, est un protocole de communication dédié à l'échange informatique de fichiers sur un réseau TCP/IP. Il intervient au niveau de la couche application du modèle OSI (couche n°7) et utilise TCP comme protocole de transport. Le protocole FTP est décrit dans la RFC 959.

FTP est un service standard d'Internet pour le transfert de fichiers. Il est important de faire la différence entre le **transfert de fichiers**, qui est réalisé par **FTP**, et l'**accès aux fichiers** à travers un réseau qui est fourni par des applications telles que **NFS** (Network File System de Sun). Le transfert de fichier consiste à recopier un fichier complet d'un système à un autre. Pour utiliser **FTP**, il faut posséder un compte sur le système distant pour pouvoir s'y connecter. Dans le cas de **FTP** anonyme, il n'y a pas besoin de compte.

FTP a été conçu dès l'origine pour fonctionner entre des machines différentes, exécutant des systèmes d'exploitation différents, utilisant des structures de fichiers différentes et éventuellement des jeux de caractères différents. Alors que **telnet** utilise un seul standard (ASCII NVT) auquel doivent se conformer les deux machines, **FTP** gère toutes les différences entre les systèmes en supportant un nombre limité de types de fichiers (ASCII, binaires, ...) et de structures de fichiers (à flux d'octets ou orientés enregistrements).

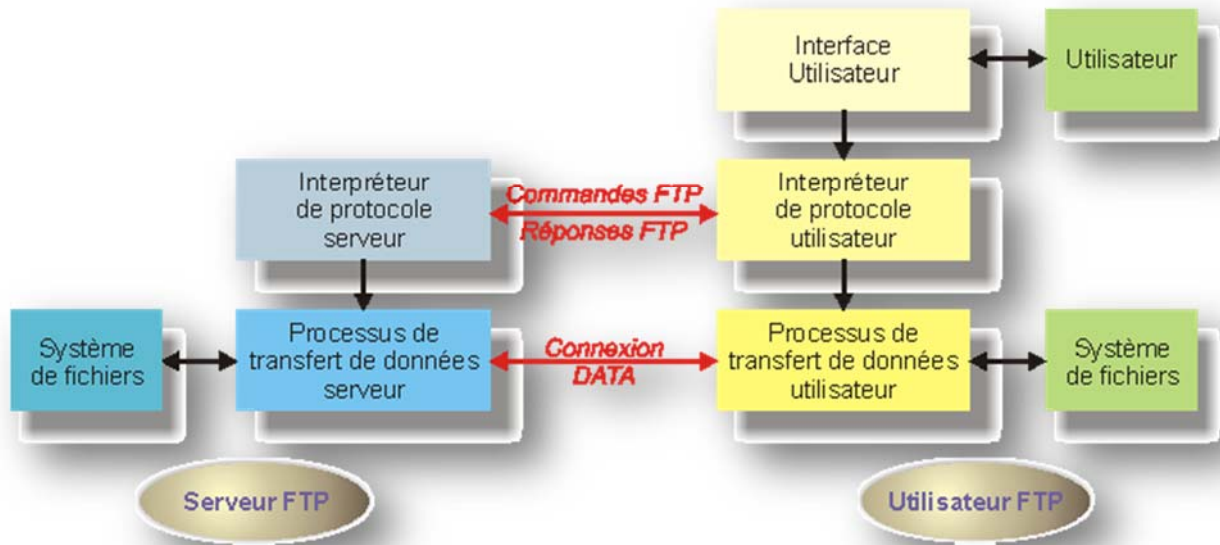
FTP est un protocole orienté connexion. C'est-à-dire qu'il y a trois étapes :

- Etablir une connexion entre le site local et le site distant
- Effectuer les traitements désirés
- Fermer la connexion

FTP utilise deux connexions TCP pour transférer un fichier :

- Une **connexion de contrôle** est utilisée pour acheminer les commandes (ou requêtes) du client vers le serveur et les réponses (ou résultats) du serveur vers le client.
- Une **connexion de transfert de données** qui est créée à chaque fois qu'un fichier est transféré entre le client et le serveur.

La figure suivante montre la configuration du client et du serveur et les deux connexions. Elle montre que l'utilisateur n'a pas à s'occuper des commandes et réponses qui sont échangées le long de la connexion de contrôle. Ces détails sont gérés par les interpréteurs de protocole. L'élément appelé « interface utilisateur » gère le type d'interface utilisé pour gérer le transfert de fichiers : interface en fenêtre, en mode ligne ou à partir d'un programme (script shell par exemple).



Pour la connexion de contrôle, le numéro de port utilisé par le serveur *ftp* est 21. C'est un port réservé (*well-known port*) qui est décrit dans le fichier */etc/services*. Le client, quand à lui obtient un numéro de port dynamiquement.

Pour la connexion de transfert de données, le numéro de port utilisé par le serveur *ftp* est 20. Le client, quand à lui, obtient un numéro de port dynamiquement.

De nombreux choix de représentation des données sont fournis par la spécification du protocole **FTP**. Les seuls qui sont encore utilisés aujourd'hui sont les types ASCII et binaires (BINARY ou IMAGE).

On utilise le mode ASCII lorsque les fichiers échangés ne contiennent que des caractères éditables. Dans ce cas, il y a prise en compte des différences entre les machines (traduction des séquences de retour à la ligne notamment). Les fichiers exécutables sont en mode binaire. En général, c'est l'utilisateur qui doit choisir le mode de transfert (par défaut c'est en général le mode ASCII qui est utilisé). Dans le doute, il vaut toujours mieux utiliser le mode binaire.

Il permet, depuis un ordinateur, de copier des fichiers depuis ou vers un autre ordinateur du réseau, d'administrer un site web, ou encore de supprimer ou modifier des fichiers sur cet ordinateur.

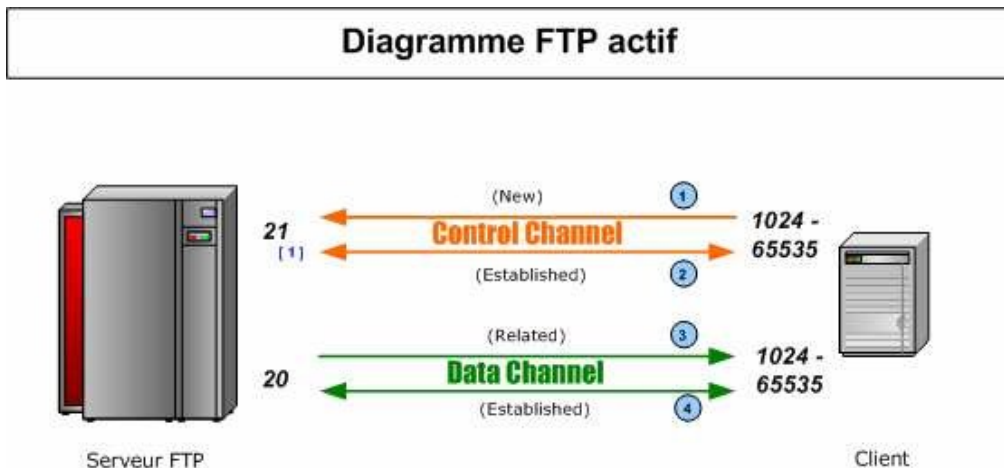
FTP obéit à un modèle client-serveur, c'est-à-dire qu'une des deux parties, le client, envoie des requêtes auxquelles réagit l'autre, appelé serveur.

En pratique, le serveur est un ordinateur sur lequel fonctionne un logiciel lui-même appelé serveur FTP, qui rend publique une arborescence de fichiers similaire à un système de fichiers Unix. Pour accéder à un serveur FTP, on utilise un logiciel client FTP (possédant une interface graphique ou en ligne de commande). Pour un OS MS-Windows, Filezilla est une référence de client FTP.

FTP utilise 2 circuits distincts au lieu d'un seul : le premier est utilisé comme canal de contrôle à travers lequel le client envoie les commandes au serveur et/ou le serveur envoie les messages de réponse au client.

Le second canal est dédié uniquement et strictement au transfert des fichiers, ce qui inclut aussi bien l'envoi de fichier au client depuis le serveur que les informations de répertoires du serveur au client.

FTP peut s'utiliser de deux façons différentes :



En mode actif, c'est le client FTP qui détermine le port de connexion à utiliser pour permettre le transfert des données. Ainsi, pour que l'échange des données puisse se faire, le serveur FTP initialisera la connexion de son port de données (port 20) vers le port spécifié par le client.

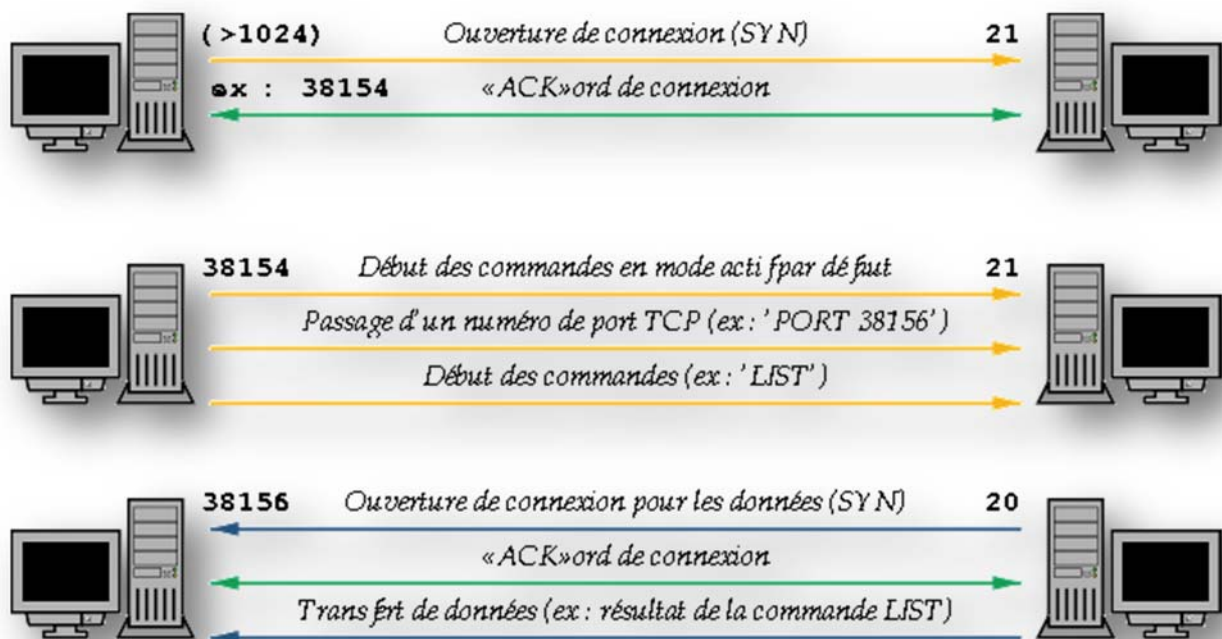
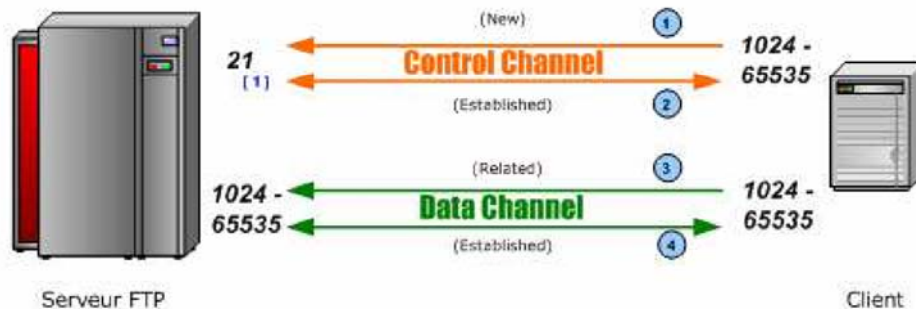
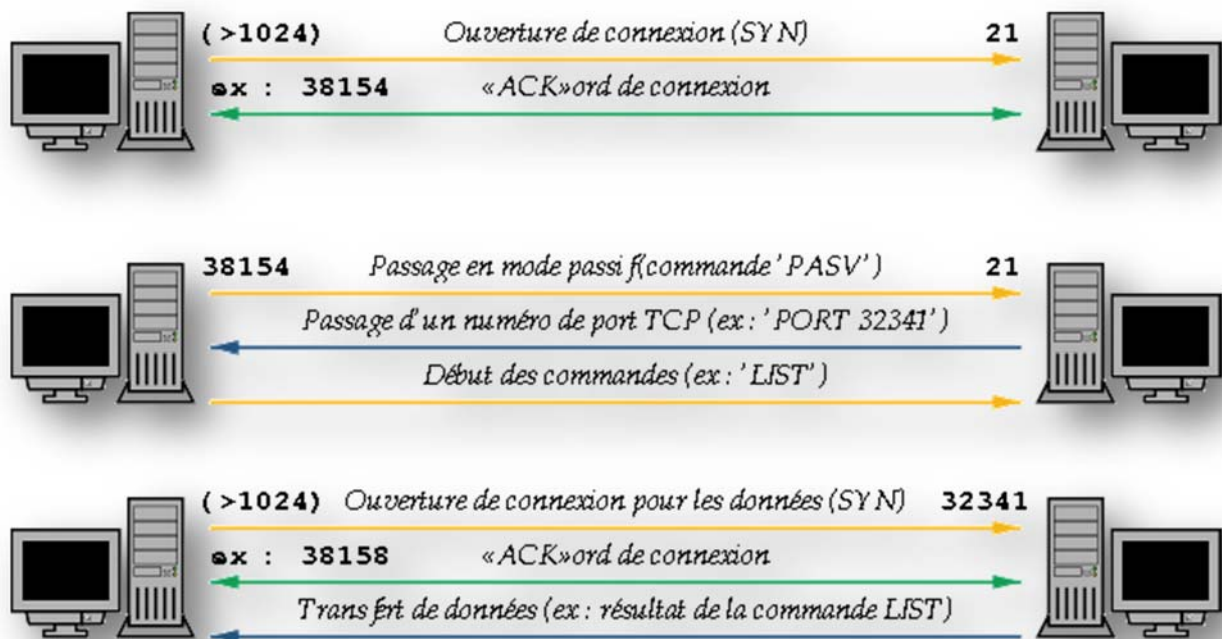


Diagramme FTP passif



En mode passif, le serveur FTP détermine lui-même le port de connexion à utiliser pour permettre le transfert des données (data connexion) et le communique au client. En cas de présence d'un pare-feu devant le serveur, celui-ci devra être configuré pour autoriser la connexion de données. L'avantage de ce mode est que le serveur FTP n'initialise aucune connexion. Ce mode fonctionne sans problèmes avec des clients derrière une passerelle NAT. Pour résumer, si l'on doit passer un pare-feu, il vaut mieux utiliser le mode passif, car le mode actif risque de se solder rapidement par un échec.



Utilisation du mode FTP

Une session FTP commence quand le client FTP démarre, un serveur FTP doit être spécifié et la connexion initialisée.

Dès que l'utilisateur s'est identifié et son mot de passe accepté par le serveur, l'utilisateur essaye généralement de localiser les fichiers qui l'intéressent et les récupère depuis le serveur sur le poste local. Dès que tous les transferts sont terminés, l'utilisateur termine sa connexion.

Le circuit de contrôle FTP reste actif pendant toute la session, en revanche, le circuit de transfert n'existe que le temps de l'envoi des données. Cette séparation en 2 circuits garantit qu'un canal restera ouvert pour transmettre les messages d'erreur si le circuit de transfert est perdu. Une autre raison de ce partitionnement est l'utilisation du transfert de fichiers en mode "stream", plutôt que d'envoyer une balise de fin de fichier, le transfert est considéré comme complet quand le circuit est fermé. Le résultat est donc que chaque fois qu'un fichier a été envoyé, le circuit de transfert se ferme et le circuit de contrôle ouvre un autre. Si une erreur provoque la fermeture du circuit de transfert, le module de contrôle considérera le fichier comme envoyé et un fragment du fichier restera dans le système de fichiers du destinataire.

FTP est fourni en standard sous diverses plateformes, dont MacOS, UNIX, Microsoft Windows, Linux, ...

La commande pour initier une session FTP est généralement la suivante:

ftp nom_serveur

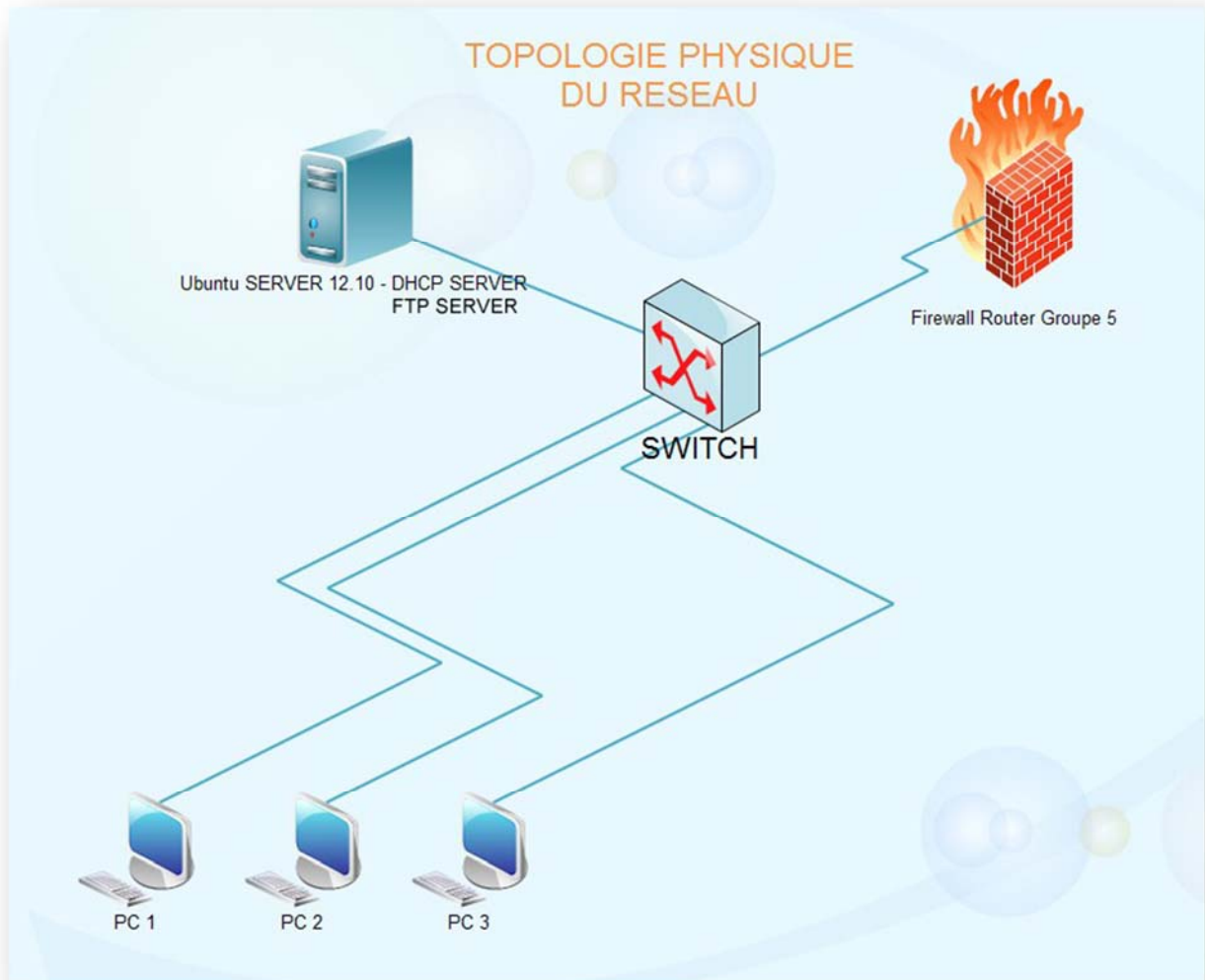
où **nom_serveur** représente le nom (ou l'adresse IP) du serveur FTP auquel on désire se connecter.

Commandes FTP :

Commande	Description
help (ou ?)	Affiche l'ensemble des commandes supportées par le serveur FTP
status	Permet de connaître certains paramètres de la machine cliente
binary	Cette commande vous fait basculer du mode ASCII (envoi de documents textes) au mode binary (envoi de fichiers en mode binaire, c'est-à-dire pour les fichiers non texte, comme des images ou des programmes)
ascii	Bascule du mode binary au mode ascii. Ce mode est le mode par défaut.
ls	Identique à la commande UNIX, mais exécutée sur le serveur FTP.
pwd	Affiche le nom répertoire courant sur le serveur FTP.
cd	Identique à la commande UNIX mais exécutée sur le serveur FTP.
lcd	Identique à la commande cd mais exécutée sur la <u>machine cliente</u> .
get nom1	Permet de récupérer le fichier nom1 présent sur le serveur FTP.
put nom1	Permet d'envoyer le fichier local nom1 sur le serveur FTP.
user	Lance une nouvelle session
mget	Identique à GET mais permet de récupérer plusieurs fichiers dont le nom est donné en paramètre.
quit	Déconnecte le logiciel client du serveur FTP et le met en état inactif
open	Ferme la session en cours et ouvre une nouvelle session sur un autre serveur FTP
close	Ferme la session en cours, en laissant le logiciel FTP client actif
!commande	Quand ! précède une commande, la commande est exécutée localement.
mkdirnom_rep	Crée le répertoire nom_rep dans le répertoire courant du serveur.
rmdirnom_rep	Supprime le répertoire nom_rep (s'il est vide) du répertoire courant du serveur.
Ren nom1 nom2	Permet de renommer le fichier nom1 en nom2 sur le serveur.
bye	Se déconnecte du serveur FTP et le met l'application FTP client en état inactif
mput	Identique à PUT mais permet d'envoyer plusieurs fichiers dont le nom est donné en paramètre.

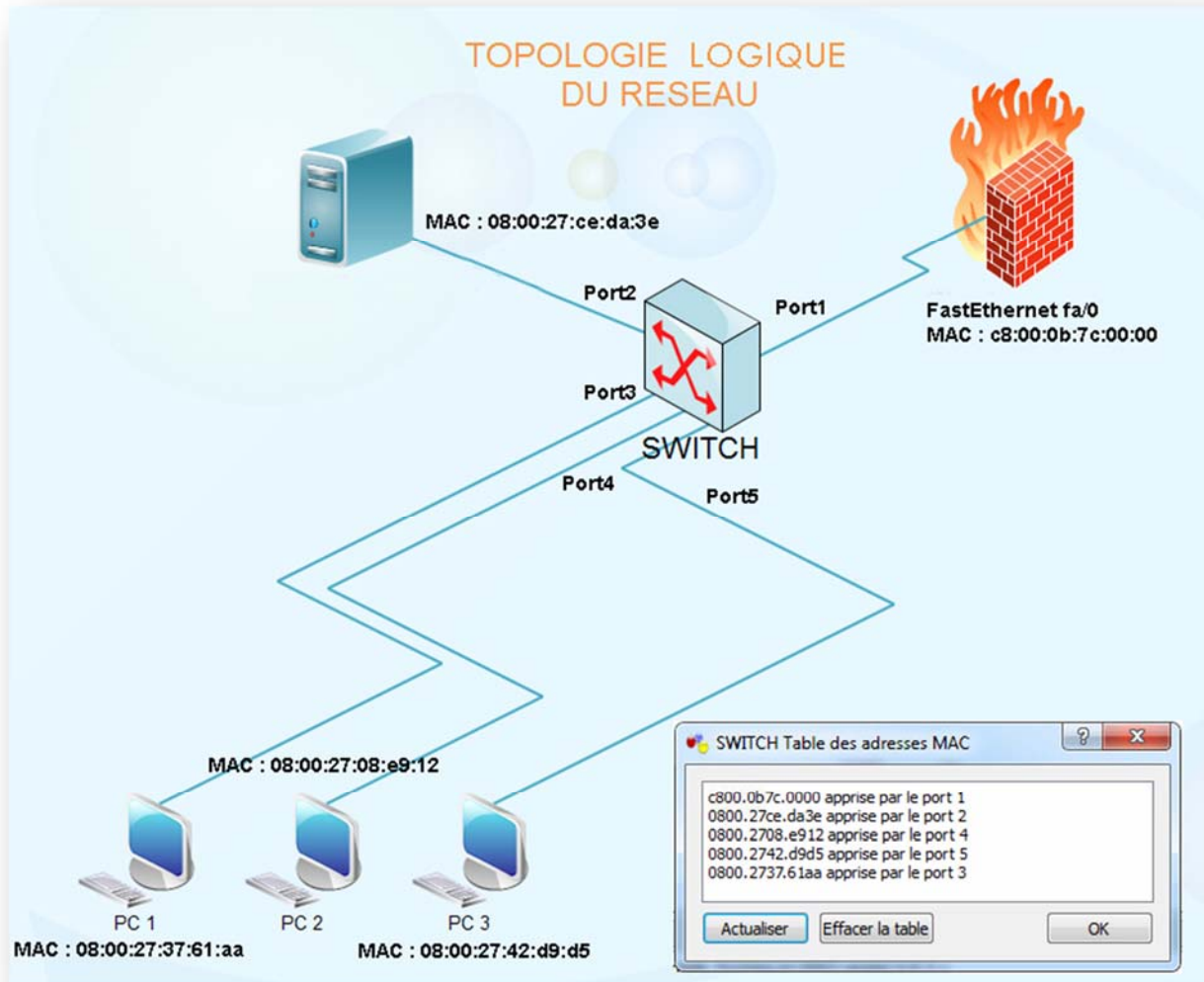
Installation du serveur FTP et configuration

1. Topologie Physique et logique du réseau



Nous disposons de 4 postes :

- 1 sera configuré comme SERVEUR FTP
- 3 Clients



2. Installer et configurer vsftpd

La création d'un serveur FTP est une étape importante pour qui veut faire du web, il existe bien d'autre système pour faire du transfert de fichier mais celui-ci est le plus simple. Ce n'est pas le plus sécurisé tout de même.

En effet un serveur FTP classique comme '**proftpd**' envoie les informations de connexion en clair! Donc si vous utiliser un réseau public, réfléchissez bien avant de vous connecter à votre serveur FTP quelqu'un de mal intentionné pourrait sniffer le réseau pour récupérer vos identifiants de connexion.

vsftpd (pour Very Secure FTP Daemon) s'annonce lui-même comme étant "probablement le plus sûr et le plus rapide des serveurs FTP pour systèmes Unix".

Il fait ce que l'on appelle du FTPS ou encore SFTP c'est-à-dire qu'il utilise le protocole SSL/TLS .

La configuration portera essentiellement sur deux usages :

- utilisation du FTP avec utilisateurs locaux
- utilisation du FTP avec des utilisateurs virtuels qui peuvent uploader

L'environnement est un serveur **Ubuntu** en version 12.04 LTS muni d'un noyau Linux en version 3.5.0 , la version de **vsftpd** est la 2.3.5

Sous Debian (Ubuntu), cela s'effectue très simplement en installant le paquetage " vsftpd ".

- Effectuer une mise à jour des paquetages disponibles en mode terminal :

```
root@ubuntu:~# apt-get update
Ign http://fr.archive.ubuntu.com quantal InRelease
Ign http://security.ubuntu.com quantal-security InRelease
Ign http://fr.archive.ubuntu.com quantal-updates InRelease
Atteint http://security.ubuntu.com quantal-security Release.gpg
Lecture des listes de paquets... Fait
```

```
root@ubuntu:/# apt-get upgrade
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les paquets suivants ont été conservés :
  linux-headers-generic linux-image-generic
Les paquets suivants seront mis à jour :
  apport apt apt-transport-https apt-utils base-files bind9 bind9-doc
  bind9-host bind9utils busybox-initramfs busybox-static coreutils curl
  dnsutils gnupg gpgv kvm libapt-inst1.5 libapt-pkg4.12 libbind9-80
  libboost-iostreams1.49.0 libcurl3 libcurl3-gnutls libdns81 libglib2.0-0
  libgnutls26 libisc83 libisc83 libisc83 libisc83 liblwres80 libnspr4 libnss3
  libpq5 libvirt-bin libvirt0 libwhoopsie0 libxenstore3.0 libxml2
  libxml2-utils linux-generic lsb-base lsb-release memtest86+ mountall perl
  perl-base perl-modules postfix postgresql-9.1 postgresql-client-9.1
  postgresql-contrib-9.1 postgresql-doc-9.1 python3-apport python3-dbus
  python3-distupgrade python3-gi python3-problem-report python3-update-manager
  python3.2 python3.2-minimal qemu-common qemu-kvm qemu-utils
  ubuntu-release-upgrader-core ufw update-manager-core vim vim-common
  vim-runtime vim-tiny whoopsie
71 mis à jour, 0 nouvellement installés, 0 à enlever et 2 non mis à jour.
Il est nécessaire de prendre 50,8 Mo dans les archives.
Après cette opération, 269 ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer [O/n] ?
```

- Vérifier que le paquetage existe bien :

```
root@ubuntu:/# apt-cache search vsftpd
vsftpd - serveur FTP léger, complet, efficace et sûr
ccze - A robust, modular log coloriser
ftpd - File Transfer Protocol (FTP) server
yasat - simple stupid audit tool
root@ubuntu:/# vsftpd--searchvs
```

- Effectuer l'installation du paquetage “ vsftpd “

```
root@ubuntu:/# apt-get install vsftpd_
```

A ce stade nous pouvons effectuer un premier essai du serveur ftp :
En mode terminal accéder au serveur ftp de la façon suivante :

```
root@Ubuntu-SERVER-64:~# ftp 172.25.205.250
Connected to 172.25.205.250.
220 (vsFTPd 2.3.5)
Name (172.25.205.250:fbernier):
```

Si vous connaissez l'adresse Ip de votre serveur, remplacer l'adresse située après FTP (172.25.205.250) par la vôtre ou mettez « **localhost** » à la place puisque pour le moment nous accédons au serveur FTP par lui-même.

A l'installation, le serveur est configuré en mode anonyme il n'y a donc pas de nom d'USER et de mot de passe à saisir.

Pour sortir du serveur taper « **exit** »

3. Configuration du serveur FTP

vsftpd se configure via le fichier **vsftpd.conf** , positionné dans **/etc** sur la majorité des distributions.

Le fichier de configuration par défaut est très restrictif, il n'autorise que les connexions anonymes, en lecture seul.

Il fait écouter le serveur sur toutes les interfaces disponibles, sur le port 21, et peut être tout à fait suffisant pour mettre en place un simple partage de fichier accessible à tous.

Voici un exemple de configuration plus complexe, qui permet d'autoriser les comptes utilisateurs présents sur le serveur à se connecter à leurs dossiers personnels, sans autoriser l'accès anonyme :

Effectuer une copie de sauvegarde du fichier **vsftpd.conf**

```
root@ubuntu:~# cp /etc/vsftpd.conf /etc/vsftpd.conf.Original
```

Edisons maintenant le fichier de configuration :

```
root@Ubuntu-SERVER-64:/# vi /etc/vsftpd.conf
```

Détails des fichiers de configuration :

Run standalone ?

[listen=YES](#)

Le mode standalone indique que le serveur est autonome, et que le service tourne en permanence.

Allow anonymous FTP ?

[anonymous_enable=no](#)

On refuse **NO** ou on accepte **YES** les connexions en mode anonyme

On autorise les connexions des utilisateurs locaux.

[local_enable=YES](#)

C'est indispensable pour que les utilisateurs virtuels (mappés sur un utilisateur local) puissent se connecter (les "vrais" utilisateurs locaux sont ensuite désactivés) avec le fichier `user_list`

Refus des commandes influant sur le système de fichier (STOR, DELE, RNFR, RNT0, MKD, RMD, APPE and SITE)

[write_enable=NO](#)

LOCAL UMASK

Le paramètre `umask` par défaut sur Debian est `022`, cela signifie que les fichiers (et les répertoires) peuvent être lus et accédés par le groupe de l'utilisateur et par tout autre utilisateur du système.

Le **umask** définit les [permissions](#) par défaut d'un [répertoire](#) ou d'un [fichier](#) créé.

Quand vous créez un fichier, par exemple avec la commande **touch**, ce fichier par défaut possède certains droits. Ce sont **666** pour un fichier (**-rw-rw-rw-**) et **777** pour un répertoire (**-rwxrwxrwx**), ce sont les droits maximum. Vous pouvez faire en sorte de changer ces paramètres par défaut. La commande **umask** est là pour ça.

Pour un fichier, si vous tapez **umask 022**, vous partez des droits maximum **666** et vous retranchez **022**, on obtient donc **644**, par défaut les fichiers auront comme droit **644** (**-rw-r--**).

Pour un répertoire, si vous tapez **umask 022**, vous partez des droits maximum **777** et vous retranchez **022**, on obtient donc **755**, par défaut les fichiers auront comme droit **644** (**-rwxr-xr-x**).

`local_umask=022`

On interdit ou autorise l'upload anonyme

`anon_upload_enable=NO`

Idem pour la création de répertoires

`anon_mkdir_write_enable=NO`

On demande à ce que les actions des utilisateurs soient "loggées"

`xferlog_enable=YES`

dirmessage_enable active l'affichage des fichiers .message à l'entrée dans les répertoires.

`dirmessage_enable=YES`

Les heures d'enregistrement des fichiers seront affichées à l'heure locale

`use_localtime=YES`

On demande à ce que les actions d' Upload et de Download des utilisateurs soient "loggées"

`xferlog_enable=YES`

On verifie que la commande PORT provienne bien du port 20 de la machine cliente


```
connect_from_port_20=YES
```

Définition du chemin du fichier où seront enregistrés les logs /var/log/vsftpd.log

```
xferlog_file=/var/log/vsftpd.log
```

On déclare les valeurs de timeout.

#Temps avant déconnexion sur une session inactive

```
idle_session_timeout=300
```

#Temps avant déconnexion sur une session active

```
data_connection_timeout=120
```

```
connect_timeout=60
```

```
accept_timeout=60
```

Par sécurité, on interdit la commande ABOR

Cette commande (*abort*) indique au serveur DTP d'abandonner tous les transferts associés à la commande précédente. Si aucune connexion de données n'est ouverte, le serveur DTP ne fait rien, sinon il la ferme. Le canal de contrôle reste par contre ouvert.

```
async_abor_enable=NO
```

Les transferts en ASCII sont souvent source de confusions

```
ascii_upload_enable=NO
```

```
ascii_download_enable=NO
```

Bannière d'accueil du site FTP

```
ftpd_banner=Bienvenue sur ce site FTP
```

On limite les utilisateurs a leur repertoire

```
chroot_local_user=YES
```

```
chroot_list_enable=NO
```

Le chroot des utilisateurs :

Il y a trois possibilités de configuration en ce qui concerne le chroot des utilisateurs. La prison est le répertoire de l'utilisateur.

1. Tous les utilisateurs sont dans une prison :

```
chroot_local_user=YES  
chroot_list_enable=NO
```

2. Seul quelques utilisateurs sont dans une prison :

```
chroot_local_user=NO  
chroot_list_enable=YES
```

3. Seul quelques utilisateurs sont "libres" :

```
chroot_local_user=YES  
chroot_list_enable=YES
```

Pour les cas 2 et 3, il vous faudra créer un fichier **/etc/vsftpd.chroot_list** contenant la liste des utilisateurs "en prison" (pour le cas 2) ou "libres" (pour le cas 3).

A ce stade, nous avons donc un serveur FTP qui autorise tous les utilisateurs locaux à accéder à leurs répertoires personnels, et qui refuse toute forme de connexion anonyme avec la configuration suivante :

1	listen=YES
2	anonymous_enable=NO
3	local_enable=YES
4	write_enable=YES
5	local_umask=022
6	anon_upload_enable=NO
7	anon_mkdir_write_enable=NO
8	dirmessage_enable=YES
9	use_localtime=YES
10	xferlog_enable=YES
11	connect_from_port_20=YES
12	xferlog_file=/var/log/vsftpd.log
13	xferlog_std_format=YES
14	idle_session_timeout=600
15	data_connection_timeout=120
16	async_abor_enable=NO
17	ascii_upload_enable=NO
18	ascii_download_enable=NO
19	chroot_local_user=YES
20	chroot_list_enable=NO

4. Utilisation du FTP avec Utilisateurs locaux

Répertoire de partage et Utilisateurs

Il peut être intéressant de mettre en place un répertoire commun aux utilisateurs, ou ils pourront accéder à des fichiers d'un dossier uniquement en lecture.

Pour cela, il faut créer un espace commun, disons "**partageftp**" :

```
root@Ubuntu-SERVER-64:/# sudo adduser partageftp
Ajout de l'utilisateur « partageftp » ...
Ajout du nouveau groupe « partageftp » (1002) ...
Ajout du nouvel utilisateur « partageftp » (1002) avec le groupe « partageftp »
...
Création du répertoire personnel « /home/partageftp »...
Copie des fichiers depuis « /etc/skel »...
Entrez le nouveau mot de passe UNIX :
Retapez le nouveau mot de passe UNIX :
passwd: password updated successfully
Changing the user information for partageftp
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Cette information est-elle correcte ? [0/n] o
root@Ubuntu-SERVER-64:/#
```

Mettez le mot de passe que vous désirez

Ne rien saisir

Puis indiquer correctement les droits :

```
root@Ubuntu-SERVER-64:/#  
root@Ubuntu-SERVER-64:/# sudo chmod -R 755 /home/partageftp  
root@Ubuntu-SERVER-64:/# sudo chown partageftp.partageftp -R /home/partageftp  
root@Ubuntu-SERVER-64:/#
```

Créer maintenant 3 utilisateurs :

- Jerome
- Oliver
- Vincent

Adduser

Créer ensuite un répertoire "/home/utilisateur/partageftp" **dans le dossier personnel de chaque utilisateur** :

```
$ mkdir /home/utilisateur/partageftp
```

```
$ sudo chown utilisateur:utilisateur /home/utilisateur/partageftp
```

```
$ sudo chmod 755 /home/utilisateur/partageftp
```

Et enfin modifier le fichier **"/etc/fstab"** pour monter automatiquement le répertoire partagé dans le home de chaque utilisateur.

Pour ce faire, il faut ajouter la ligne suivante dans le **fstab**, une ligne par utilisateur différent :

/home/partage /home/utilisateur/partage auto bind,defaults 0 0

```
root@Ubuntu-SERVER-64:/# vi /etc/fstab_  
  
# /etc/fstab: static file system information.  
#  
# Use 'blkid' to print the universally unique identifier for a  
# device; this may be used with UUID= as a more robust way to name devices  
# that works even if disks are added and removed. See fstab(5).  
#  
# <file system> <mount point> <type> <options> <dump> <pass>  
proc /proc proc nodev,noexec,nosuid 0 0  
# / was on /dev/sda1 during installation  
UUID=7100077e-9efa-41b2-a831-1c93c7b37b7c / ext4 errors=remount  
-ro 0 1  
# swap was on /dev/sda5 during installation  
UUID=940e1eb1-fa49-456e-b33b-f8af63171a2a none swap sw  
0 0  
/dev/fd0 /media/floppy0 auto rw,user,noauto,exec,utf8 0 0  
/home/partageftp /home/jerome/partageftp auto bind,defaults 0 0  
/home/partageftp /home/oliver/partageftp auto bind,defaults 0 0  
/home/partageftp /home/vincent/partageftp auto bind,defaults 0 0
```

Monter ensuite le partage créé pour chacun des utilisateurs avec la commande :

mount /home/utilisateur/partageftp (pour les 3 utilisateurs dans notre cas)

Tous les utilisateurs disposent donc d'un répertoire "**partageftp**" commun.

```
root@Ubuntu-SERVER-64:/home# tree
.
├── fbernier
│   └── netstat.txt
├── jerome
│   └── partageftp
├── oliver
│   └── partageftp
├── partageftp
├── vincent
│   └── partageftp
└──
```

B directories, 1 file
root@Ubuntu-SERVER-64:/home#

A ce stade vous pouvez enfin tester votre serveur FTP sur un client à l'aide d'un client FTP comme « **FileZilla** »

```
Statut : Connexion à 172.25.205.250:21...
Statut : Connexion établie, attente du message d'accueil...
Réponse : 220 Bienvenue sur cet excellent site de FTP.
Commande : USER jerome
Réponse : 331 Please specify the password.
Commande : PASS ****
Réponse : 500 OOPS: vsftpd: refusing to run with writable root inside chroot()
Erreur : Erreur critique
Erreur : Impossible d'établir une connexion au serveur
```

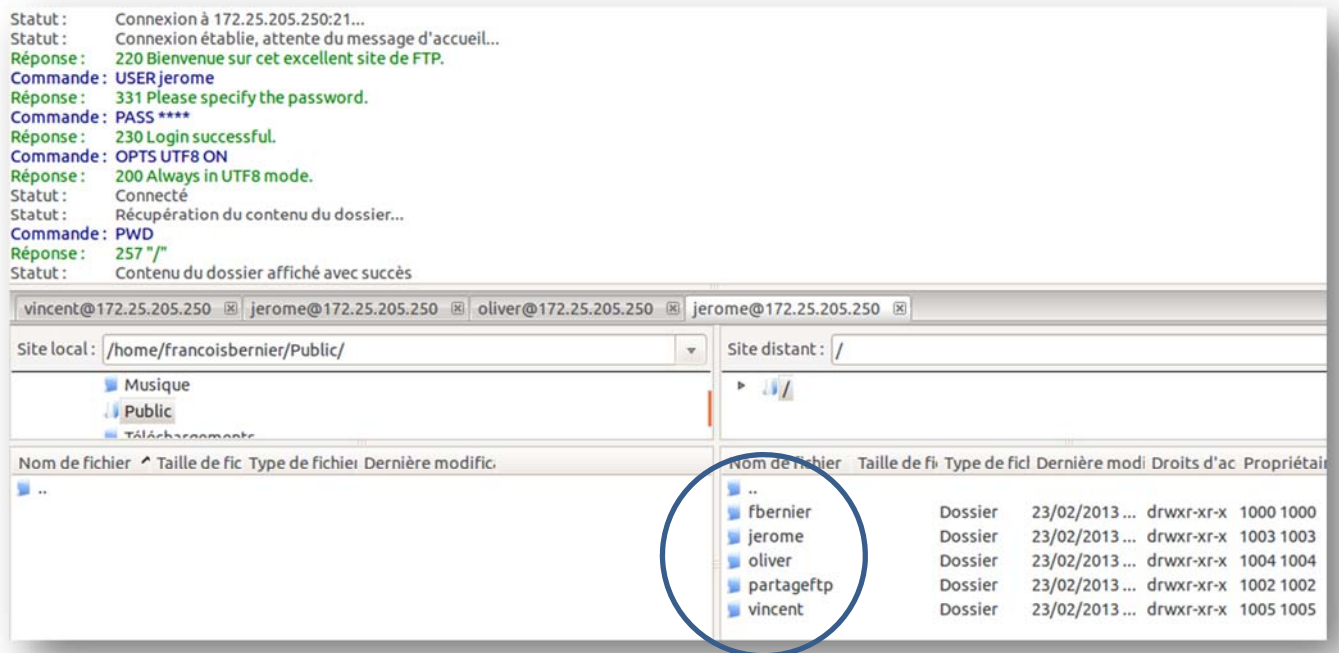
Et vous avez une grande chance de voir apparaître l'erreur suivante :

500 OOPS: vsftpd: refusing to run with writable root inside chroot()

Il s'agit en fait d'un bug dans la version 2.3.5 de **vsftpd** qui empêche un utilisateur de se « logger » avec le répertoire racine accessible en écriture à cause de vulnérabilités possibles.

On pourrait dévier le bug en fixant le répertoire d'accès avec la commande « **local_root=/home** » dans le fichier **vsftpd.conf** , dans ce cas l'accès au site FTP

fonctionne , mais l'utilisateur connecté aura accès à tous le répertoire **/home** , donc à tous les répertoire utilisateur.



2 solutions s'offrent à vous pour contrer ce problème :

- soit réinstaller une version ultérieure de **vsftpd** qui ne contient pas ce bug.
- Soit d'effectuer une mise à jour de la version de **vsftpd** incluant une option autorisant l'écriture (la version 3.0.0 de **vsftpd** corrige ce bug , mais n'a toujours pas été intégrée aux mises à jour disponible pour Ubuntu 12....)

La procédure à suivre est la suivante :

Installer le lien vers la version corrigée et réinstaller **vsftpd**.

```
sudo add-apt-repository ppa:thefrontiergroup/vsftpd
```

```
sudo apt-get update
```

```
sudo apt-get install vsftpd
```

Il faudra effectuer les modifications suivantes dans le fichier **vsftpd.conf**

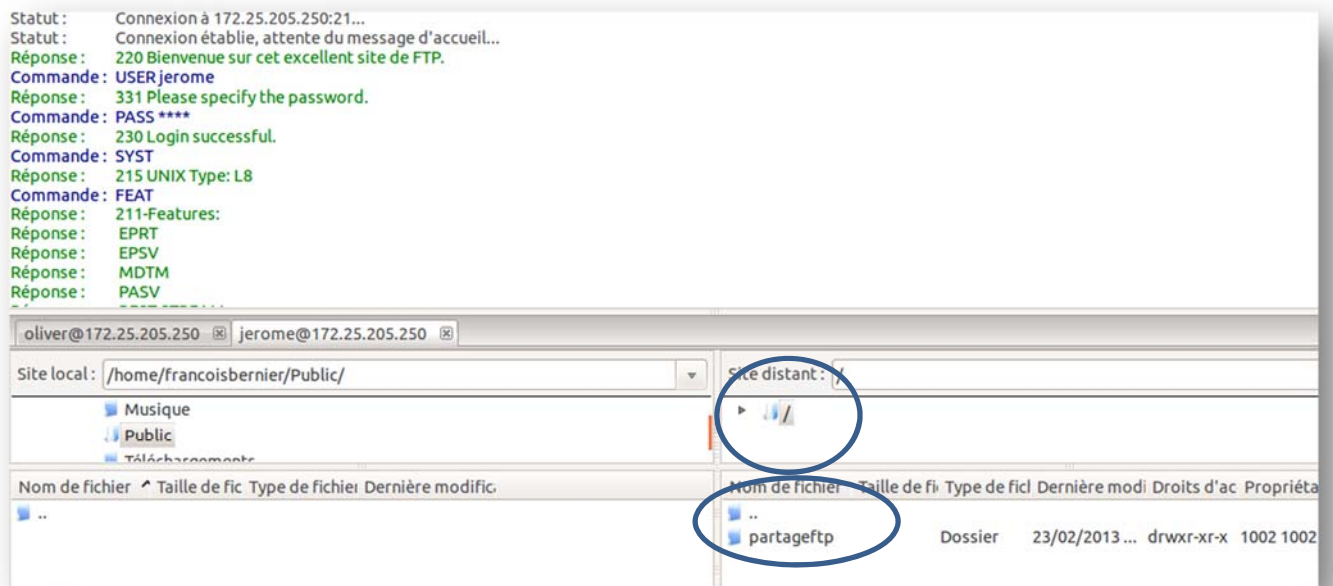

```
chroot_local_user=YES
chroot_list_enable=NO
#local_root=/home

user_sub_token=$USER
local_root=/home/$USER
allow_writeable_chroot=YES
```

Redémarrer le serveur ftp

```
root@Ubuntu-SERVER-64:/home/fbernier# sudo service vsftpd restart
vsftpd stop/waiting
vsftpd start/running, process 1698
```

Tester avec FileZilla une connexion du poste client vers le serveur avec un utilisateur



La connexion fonctionne correctement et l'utilisateur est bien « emprisonné » dans son dossier.

Effectuer une copie d'un document quelconque dans le répertoire **/home/partageftp** et ensuite vérifier que ce fichier est bien présent dans le **partageftp** de tous les autres utilisateurs avec la commande **tree** :

```
root@Ubuntu-SERVER-64:/home# tree
.
├── fbernier
│   └── netstat.txt
├── jerome
│   └── partageftp
│       └── Abstraction-Fond-décran-des-graphiques-lespace.jpg
├── oliver
│   └── partageftp
│       └── Abstraction-Fond-décran-des-graphiques-lespace.jpg
├── partageftp
│   └── Abstraction-Fond-décran-des-graphiques-lespace.jpg
└── vincent
    └── partageftp
        └── Abstraction-Fond-décran-des-graphiques-lespace.jpg

8 directories, 5 files
root@Ubuntu-SERVER-64:/home# _
```

Le partage fonctionne, le fichier apparaît bien présent dans le répertoire de partage de chaque utilisateur enregistré.

Pour aller plus loin, il peut aussi être intéressant de disposer d'un répertoire commun accessible en écriture.

Pour cela, il suffit de créer un répertoire dans le "home" de l'utilisateur "partage", et de lui donner les droits adéquats:

```
$ sudo mkdir /home/partageftp/upload
```

```
$ sudo chown partageftp:partageftp /home/partageftp/upload
```

```
$ sudo chmod 777 /home/partageftp/upload
```

On obtient arborescence suivante:

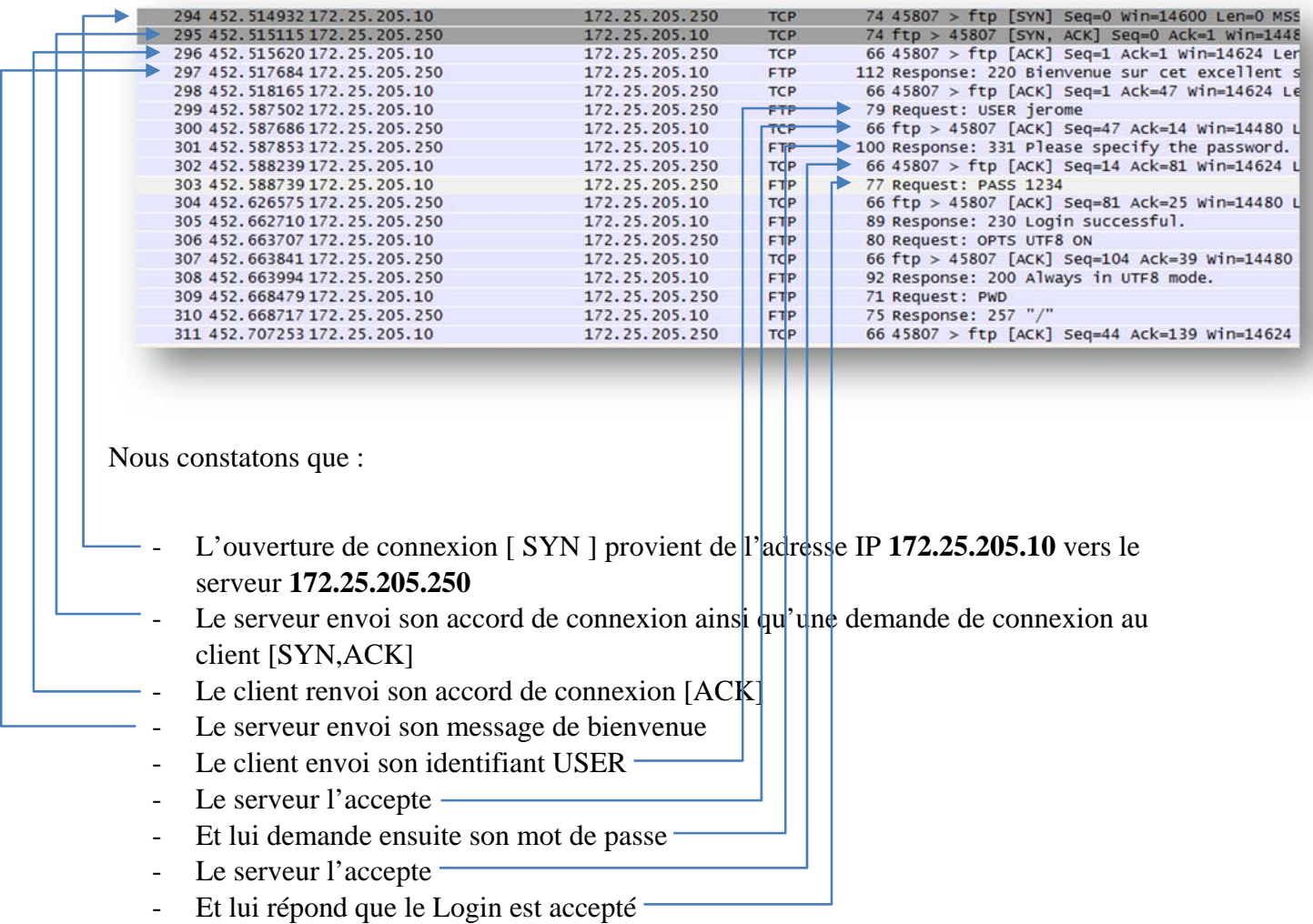
```
root@Ubuntu-SERVER-64:/home# tree
.
├── fbernier
│   └── netstat.txt
├── jerome
│   └── partageftp
│       ├── Abstraction-Fond-décran-des-graphiques-lespace.jpg
│       └── upload
├── oliver
│   └── partageftp
│       ├── Abstraction-Fond-décran-des-graphiques-lespace.jpg
│       └── upload
├── partageftp
│   ├── Abstraction-Fond-décran-des-graphiques-lespace.jpg
│   └── upload
└── vincent
    └── partageftp
        ├── Abstraction-Fond-décran-des-graphiques-lespace.jpg
        └── upload

12 directories, 5 files
root@Ubuntu-SERVER-64:/home#
```

Les utilisateurs disposent donc d'un répertoire d'échange en lecture seule et d'un second en écriture.

Exploration des Trames avec le logiciel Wireshark

Voici ci-dessous la capture d'une connexion TCP non sécurisée par le logiciel Wireshark



No.	Time	Source	Destination	Protocol	Length	Info
294	452.514932	172.25.205.10	172.25.205.250	TCP	74	45807 > ftp [SYN] Seq=0 win=14600 Len=0 MSS
295	452.515115	172.25.205.250	172.25.205.10	TCP	74	ftp > 45807 [SYN, ACK] Seq=0 Ack=1 win=1448
296	452.515620	172.25.205.10	172.25.205.250	TCP	66	45807 > ftp [ACK] Seq=1 Ack=1 win=14624 Len
297	452.517684	172.25.205.250	172.25.205.10	FTP	112	Response: 220 Bienvenue sur cet excellent s
298	452.518165	172.25.205.10	172.25.205.250	TCP	66	45807 > ftp [ACK] Seq=1 Ack=47 win=14624 Le
299	452.587502	172.25.205.10	172.25.205.250	FTP	79	Request: USER jerome
300	452.587686	172.25.205.250	172.25.205.10	TCP	66	ftp > 45807 [ACK] Seq=47 Ack=14 win=14480 L
301	452.587853	172.25.205.250	172.25.205.10	FTP	100	Response: 331 Please specify the password.
302	452.588239	172.25.205.10	172.25.205.250	TCP	66	45807 > ftp [ACK] Seq=14 Ack=81 win=14624 L
303	452.588739	172.25.205.10	172.25.205.250	FTP	77	Request: PASS 1234
304	452.626575	172.25.205.250	172.25.205.10	TCP	66	ftp > 45807 [ACK] Seq=81 Ack=25 win=14480 L
305	452.662710	172.25.205.250	172.25.205.10	FTP	89	Response: 230 Login successful.
306	452.663707	172.25.205.10	172.25.205.250	FTP	80	Request: OPTS UTF8 ON
307	452.663841	172.25.205.250	172.25.205.10	TCP	66	ftp > 45807 [ACK] Seq=104 Ack=39 win=14480
308	452.663994	172.25.205.250	172.25.205.10	FTP	92	Response: 200 Always in UTF8 mode.
309	452.668479	172.25.205.10	172.25.205.250	FTP	71	Request: PWD
310	452.668717	172.25.205.250	172.25.205.10	FTP	75	Response: 257 "/"
311	452.707253	172.25.205.10	172.25.205.250	TCP	66	45807 > ftp [ACK] Seq=44 Ack=139 win=14624

Nous constatons que :

- L'ouverture de connexion [SYN] provient de l'adresse IP **172.25.205.10** vers le serveur **172.25.205.250**
- Le serveur envoie son accord de connexion ainsi qu'une demande de connexion au client [SYN,ACK]
- Le client renvoie son accord de connexion [ACK]
- Le serveur envoie son message de bienvenue
- Le client envoie son identifiant USER
- Le serveur l'accepte
- Et lui demande ensuite son mot de passe
- Le serveur l'accepte
- Et lui répond que le Login est accepté

Les ports de communications :

```
Internet Protocol Version 4, Src: 172.25.205.10 (172.25.205.10), Dst: 172.25.205.250 (172.25.205.250)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
  Total Length: 60
  Identification: 0xf929 (63785)
  Flags: 0x02 (Don't Fragment)
  Fragment offset: 0
  Time to live: 64
  Protocol: TCP (6)
  Header checksum: 0x4e5a [correct]
  Source: 172.25.205.10 (172.25.205.10)
  Destination: 172.25.205.250 (172.25.205.250)
  Transmission Control Protocol, Src Port: 45807 (45807), Dst Port: ftp (21), Seq: 0, Len: 0
    Source port: 45807 (45807)
    Destination port: ftp (21)
    [Stream index: 13]
    Sequence number: 0 (relative sequence number)
    Header length: 40 bytes
    Flags: 0x002 (SYN)
    window size value: 14600
    [Calculated window size: 14600]
    Checksum: 0x5e9d [validation disabled]
    Options: (20 bytes)
```

- Le client envoi sa demande [SYN] par le port 45807 vers le port 21 du serveur

Toutes les communications de validation de connexion s'effectuent sur ces mêmes ports.

Configuration avec Utilisateurs virtuels

Les utilisateurs sont capables de tout... une protection supplémentaire de **vsftpd** consiste à faire en sorte que le client qui accède ainsi à votre serveur aient des pouvoirs très limités...

L'idée consiste à créer un utilisateur très particulier, un utilisateur virtuel, qui n'aura donc de droits que dans le cadre de **vsftpd**.

Puisqu'il n'existe pas vraiment sur le système (il n'a pas de mot de passe sur la machine elle-même), il ne pourra faire autre chose que... ce que vous voudrez bien qu'il fasse en tant qu'utilisateur de votre serveur **FTP** : lire, bien sûr, mais pourquoi pas aussi écrire ou créer des fichiers.

Dès qu'il sortira des répertoires que vous aurez autorisés pour lui (en tentant par exemple d'écrire ailleurs, d'installer un programme ou de lire autre chose), donc dès qu'il tentera d'échapper au cadre du serveur **vsftpd**, le système le rejettera en tant qu'utilisateur inconnu ! Il s'agit là d'une protection encore plus efficace.

Elle s'opère en quatre étapes : la création d'une micro base de données qui contient les utilisateurs que vous autoriserez, la liaison de PAM avec cette base, la création d'un utilisateur virtuel, vers lequel sera mappé tout utilisateur autorisé et le paramétrage de **vsftpd** pour lui donner les droits que vous voudrez.

Création de la base de données

Il s'agit là d'une étape délicate si vous ne maîtrisez pas PAM.

Pour faire simple, PAM est le module d'authentification le plus efficace et le plus développé à ce jour sur les systèmes Linux.

Il utilise une base de données qui contient la liste des utilisateurs. Bien sûr, nous pourrions créer un utilisateur et **PAM** l'intégrerait directement dans sa base de données. Mais justement, nous ne voulons pas que l'utilisateur existe vraiment sur le système !

Nous ne le créons donc que pour PAM et en dehors de la machine elle-même.

Pour ce faire, créons un fichier, qui contiendra la liste des utilisateurs virtuels que nous voulons ajouter.

La règle est : sur la première ligne un login, sur la seconde le mot de passe correspondant, sur la troisième un autre login, sur la quatrième son mot de passe, etc. autant de fois que vous voudrez ajouter d'utilisateurs.

Par exemple, créons le fichier `virtuels.txt`, qui contiendra :

Nous avons besoin d'installer le paquet "libpam-pwdfilere" s'il n'est pas présent sur le serveur:

apt-get install libpam-pwdfilere

Pour utiliser PAM vous devez créer un nouveau fichier **"/etc/pam.d/vsftpd"** et y ajouter les lignes suivantes :

D'abord sauvegarder votre fichier d'origine : **cp /etc/pam.d/vsftpd /etc/pam.d/vsftpd.bak**

```
auth    required    pam_pwdfilere.so pwdfilere /etc/vsftpd/ftpd.passwd
account required    pam_permit.so
```

Il faut ensuite prendre soin de commenter ou retirer toutes les autres lignes présentes dans le fichier **vsftpd.conf** qui doit uniquement contenir les lignes ci-dessous :

Attention : avec cette configuration, les utilisateurs locaux ne seront plus capables de se connecter.

listen=YES

anonymous_enable=NO

local_enable=YES

write_enable=YES

local_umask=022

nopriv_user=vsftpd

virtual_use_local_privs=YES

guest_enable=YES

user_sub_token=\$USER

local_root=/var/www/\$USER ←

chroot_local_user=YES

hide_ids=YES

guest_username=vsftpd

Le nom du répertoire « **local_root** » sera celui où vous créerez vos utilisateurs

Créer un utilisateur local sans shell

On nomme notre utilisateur '**vsftpd**'.

Lorsqu'un utilisateur virtuel se connectera, Ubuntu utilisera ce nouvel utilisateur '**vsftpd**' pour ses droits d'accès et de possession: **chown** et **chmod**.

```
useradd --home /home/vsftpd --gid nogroup -m --shell /bin/false vsftpd
```

Il faut maintenant créer des utilisateurs virtuels :

```
htpasswd -cd /etc/vsftpd.passwd utilisateur1
```

Pour les utilisateurs suivants :

```
htpasswd -d /etc/vsftpd.passwd utilisateurX
```

Puis les répertoires de ces utilisateurs :

```
mkdir /var/www/utilisateurX
```

```
chmod 755 /var/www/utilisateurX
```

```
root@Ubuntu-SERVER-64:/# htpasswd -c /etc/vsftpd.passwd jerome1
New password:
Re-type new password:
htpasswd: password verification error
root@Ubuntu-SERVER-64:/# htpasswd -c /etc/vsftpd.passwd oliver1
New password:
Re-type new password:
Adding password for user oliver1
root@Ubuntu-SERVER-64:/# htpasswd -c /etc/vsftpd.passwd vincent1
New password:
Re-type new password:
Adding password for user vincent1
root@Ubuntu-SERVER-64:/# mkdir /var/www/jerome1
root@Ubuntu-SERVER-64:/# mkdir /var/www/oliver1
root@Ubuntu-SERVER-64:/# mkdir /var/www/vincent1
root@Ubuntu-SERVER-64:/# chmod 755 /var/www/jerome1
root@Ubuntu-SERVER-64:/# chmod 755 /var/www/oliver1
root@Ubuntu-SERVER-64:/# chmod 755 /var/www/vincent1
```


Comme précédemment on crée un répertoire `"/var/www/utilisateur/partageftp"` dans le dossier personnel de chaque utilisateur :

```
$ mkdir /var/www/utilisateur/partageftp
```

```
$ sudo chown utilisateur:utilisateur /var/www/utilisateur/partageftp
```

```
$ sudo chmod 755 /var/www/utilisateur/partageftp
```

Et enfin modifier le fichier `"/etc/fstab"` pour monter automatiquement le répertoire partagé dans le home de chaque utilisateur.

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
proc /proc proc nodev,noexec,nosuid 0 0
# / was on /dev/sda1 during installation
UUID=7100077e-9efa-41b2-a831-1c93c7b37b7c / ext4 errors=remount
-ro 0 1
# swap was on /dev/sda5 during installation
UUID=940e1eb1-fa49-456e-b33b-f8af63171a2a none swap sw
0 0
/dev/fd0 /media/floppy0 auto rw,user,noauto,exec,utf8 0 0
/home/partageftp /home/jerome/partageftp auto bind,defaults 0 0
/home/partageftp /home/oliver/partageftp auto bind,defaults 0 0
/home/partageftp /var/www/jerome1/partageftp auto bind,defaults 0 0
/home/partageftp /var/www/oliver1/partageftp auto bind,defaults 0 0
/home/partageftp /var/www/vincent1/partageftp auto bind,defaults 0 0
```

Pour ce faire, il faut ajouter la ligne suivante dans le **fstab**, une ligne par utilisateur différent :

```
/home/partage /var/www/utilisateur/partage auto bind,defaults 0 0
```

Et enfin monter le dossier de partage pour chacun des 3 nouveaux utilisateurs avec la commande :

```
mount /var/www/utilisateur/partageftp ( pour les 3 nouveaux utilisateurs )
```

Si nous vérifions l'arborescence de notre dossier `/var/www`, nous constatons que les répertoires utilisateur dispose de leurs dossiers `/partageftp` (en lecture seule) et le répertoire `/upload` (en écriture) créé précédemment .

```
root@Ubuntu-SERVER-64:/var/www# tree
.
├── index.html
├── jerome1
│   ├── partageftp
│   │   ├── Abstraction-Fond-décran-des-graphiques-lespace.jpg
│   │   └── upload
│   │       ├── firefox-81-firefox-informatique-small.jpg
│   │       └── M31_hallas.jpg
├── oliver1
│   ├── partageftp
│   │   ├── Abstraction-Fond-décran-des-graphiques-lespace.jpg
│   │   └── upload
│   │       ├── firefox-81-firefox-informatique-small.jpg
│   │       └── M31_hallas.jpg
└── vincent1
    ├── partageftp
    │   ├── Abstraction-Fond-décran-des-graphiques-lespace.jpg
    │   └── upload
    │       ├── firefox-81-firefox-informatique-small.jpg
    │       └── M31_hallas.jpg

9 directories, 10 files
root@Ubuntu-SERVER-64:/var/www# _
```

Administration Réseau sous Ubuntu SERVER 12.10-Serveur FTP

Un essai de connexion avec un des utilisateurs virtuel, permet de constater la bonne arborescence des répertoires, les droits des dossiers **/partage** et **/upload** et que l'utilisateur virtuel est bien bloqué dans son dossier.

