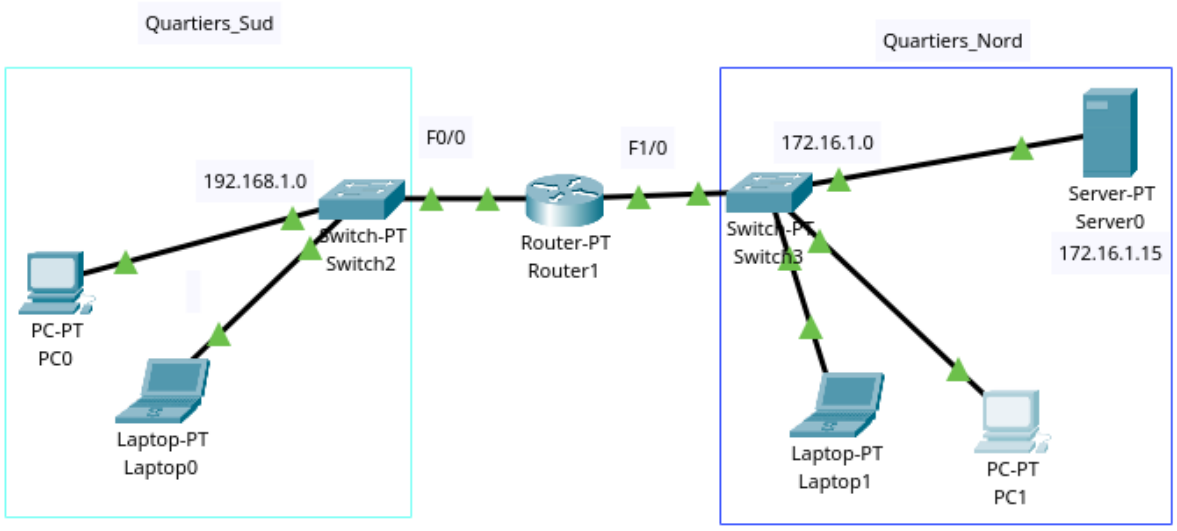
Job 01:

On commence par disposer le matériel, puis on passe directement au Job 02.

Job 02:

On place les Switch de part et d'autre du routeur, un sur l'interface FastEthernet0/0 et l'autre sur l'interface FatEthernet1/0
On assigne à chaque interface une ip: ici 192.168.1.1 et 172.16.1.1 avec un subnet mask de 255.255.255.0 à chaque fois.

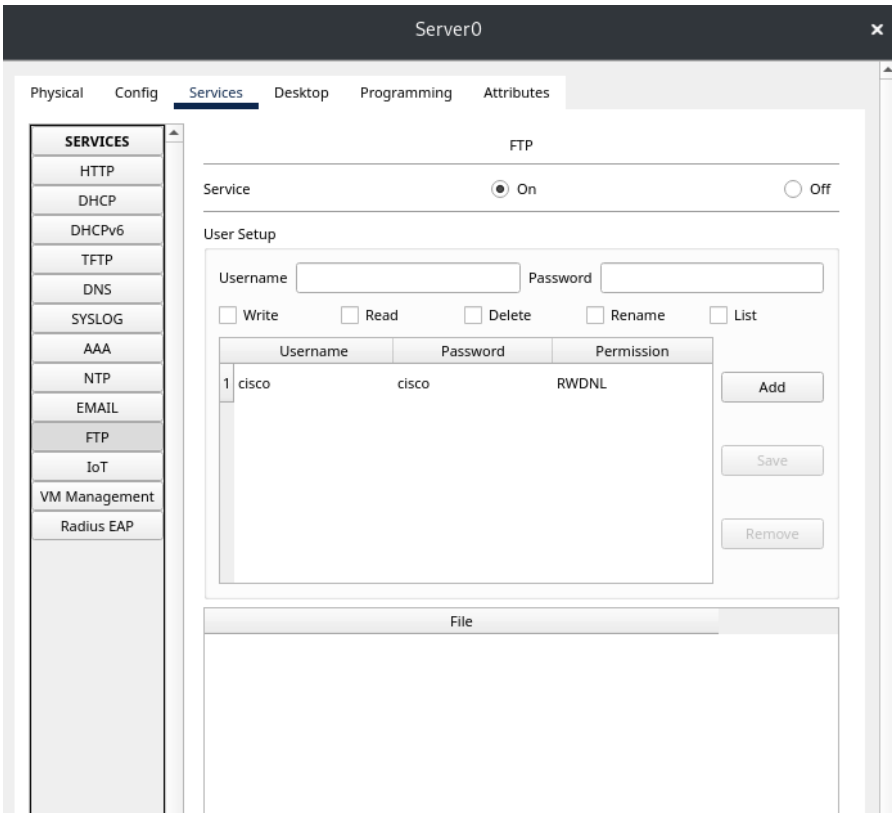
On configure chaque interface comme une pool d'IP données par le DHCP.



On configure chaque ordinateur pour que son adresse IP soit donnée par le DHCP, et on donne l'IP statique au Serveur: 172.16.1.15
Notons que comme c'est normalement le DHCP qui s'occupe d'indiquer un DNS et une Gateway au périphérique, ici le Serveur n'en a pas. Je le renseigne donc manuellement: 172.16.1.2 pour le DNS et 172.16.1.1 pour la Default Gateway.

Job 03:

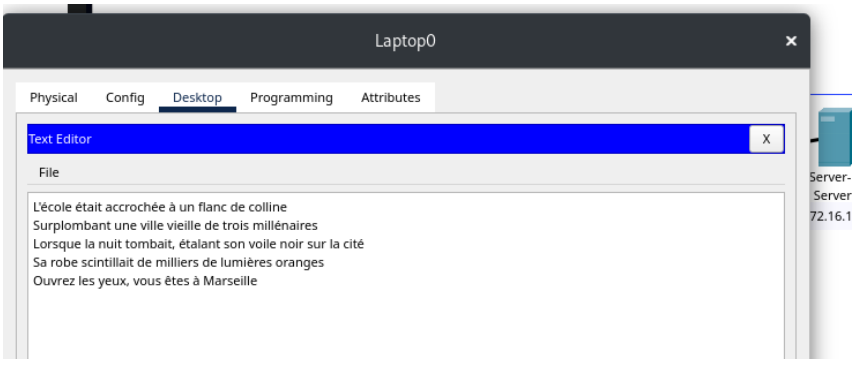
Pour configurer le serveur FTP, on se rend dans cet onglet:



Ici, on voit que le service est déjà sur On, et qu'un compte est déjà créé avec le nom d'utilisateur cisco et le mot de passe cisco.
De plus, on voit que les permissions sont entières: Read, Write, Delete, Rename, List (RWDNL)

Job 04:

Pour créer le fichier texte, on se rend dans le Desktop d'un des ordinateurs du réseau Quartiers_Sud, puis dans "Text Editor".
On écrit le texte voulu puis on appuie sur Ctrl-S et on nomme le fichier "mon_test.txt".



Pour l'envoyer au serveur FTP, on ouvre le command prompt et on y écrit:
ftp 172.16.1.15 (l'adresse ip du serveur)
On rentre ensuite notre nom d'utilisateur ainsi que notre mot de passe.

```
ftp 172.16.1.15
Trying to connect...172.16.1.15
Connected to 172.16.1.15
220- Welcome to PT Ftp server
Username:cisco
331- Username ok, need password
Password:
230- Logged in
(passive mode On)
ftp>
```

Dans cette interface, on peut utiliser le point d'interrogation pour obtenir les différentes commandes possibles:

```
ftp>?
         ?
         cd
         delete
         dir
         get
         help
         passive
         put
         pwd
         quit
         rename
ftp>
```

Pour envoyer le fichier, on utilise donc la commande "put mon_test.txt"

```
ftp>put mon_test.txt

Writing file mon_test.txt to 172.16.1.15:
File transfer in progress...

[Transfer complete - 260 bytes]

260 bytes copied in 0.042 secs (6190 bytes/sec)
ftp>
```

Ici, le command prompt nous indique donc que le transfert du fichier s'est bien effectué, que le fichier pèse 260 octets, qui ont été transférés en 0.042 secondes.
On exécute la commande "quit" pour quitter l'interface ftp.

On se rend ensuite dans le command prompt d'un autre ordinateur, cette fois-ci sur le réseau Quartiers_Nord par exemple.
On utilise le même procédé que pour l'envoi, mais avec la commande "get mon_test.txt" cette fois-ci.

```
C:\>ftp 172.16.1.15
Trying to connect...172.16.1.15
Connected to 172.16.1.15
220- Welcome to PT Ftp server
Username:cisco
331- Username ok, need password
Password:
230- Logged in
(passive mode On)
ftp>get mon_test.txt

Reading file mon_test.txt from 172.16.1.15:
File transfer in progress...

[Transfer complete - 260 bytes]

260 bytes copied in 0 secs
ftp>
```

L'opération dans le sens inverse fonctionne de la même manière.

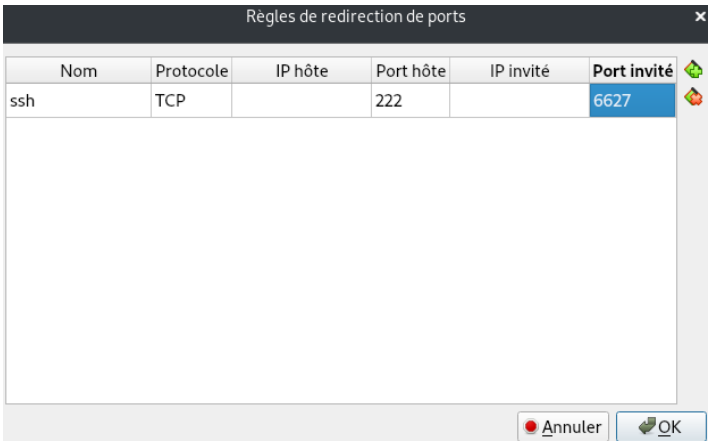
Job 06:

Je commence par créer la machine virtuelle, donne les droits de superutilisateur à l'utilisateur que j'utilise, puis installe le service "openssh-server".
Notons que nous pourrions procéder à la suite directement, mais par bonne pratique, j'effectue deux ou trois opérations de sécurisation (basique) du serveur:
Tout d'abord, dans la configuration du ssh, on peut modifier le port utilisé pour la connexion en ssh, puis la ligne PermitRootLogin désactiver la connexion en root directement (permettant ainsi d'éviter qu'un utilisateur malveillant se connecte en root puis ait l'accès total sur le serveur, ses groupes et permissions, par exemple).

```
  GNU nano 7.2                              /etc/ssh/sshd_config *
# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

Port 6627
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
```

Ensuite, on peut configurer un firewall basique, avec ufw ou iptables, en autorisant uniquement les connexions sur le port ssh (22 de base, 6627 maintenant) et ftp (21).
Enfin, on peut configurer un Fail2Ban ainsi qu'un système de logs qui notifie l'administrateur lorsque quelqu'un se connecte ou tente de se connecter.
Bref, ces opérations ne sont pas obligatoires mais sont de bonnes pratiques à prendre pour une future carrière en administration système/cybersécurité.

Enfin (et cette opération sera à répéter pour le port 21 pour le FTP), on doit, dans la configuration de notre VM, préciser une redirection de port: ceci sert à faire comprendre à notre ordinateur que pour accéder au port 6627 de notre VM, celui-ci devra passer par le port 222 (par exemple).



Maintenant que ceci est fait, on démarre le serveur/redémarre le serveur:
sudo systemctl status ssh (pour vérifier le statut du service)
sudo systemctl start ssh (pour le démarrer)
sudo systemctl restart ssh (pour le redémarrer)
sudo systemctl enable ssh (pour préciser que l'on veut démarrer le service en même temps que le serveur lui-même)

Pour se connecter en ssh au serveur (la VM): sudo ssh -p 222 utilisateur@localhost

Job 07:

On effectue: **sudo apt update && sudo apt upgrade** (bonne pratique à prendre lorsque l'on installe de nouveaux paquets)
Puis on installe proftpd avec: sudo apt install proftpd-basic
On vérifie son statut de la même façon que pour le ssh avec systemctl:
sudo systemctl status proftpd

On crée un backup du fichier de configuration:
sudo cp /etc/proftpd/proftpd.conf /etc/proftpd/proftpd.conf.bak
Puis on ouvre le fichier de configuration:
sudo nano /etc/proftpd/proftpd.conf

Ici, on désactive l'IPv6 puisqu'on ne va pas s'en servir (on se connectera en IPv4 au serveur), on peut aussi modifier le port de connexion comme on l'a fait avec le ssh.

On peut déterminer la durée pendant laquelle on n'interagit pas avec le service ftp avant que le serveur ne déclare un timeout.
On peut, enfin (le plus important pour nous ici), déclarer dans quel répertoire chaque utilisateur atterrira lorsqu'il se connectera:
DefaultRoot / merry signifiera que Merry se connectera à la racine de l'arborescence et aura donc accès à tout le système, tandis que:
DefaultRoot /home/pippin pippin signifiera que pippin n'a accès qu'à son répertoire utilisateur
Plus simplement, on peut ne pas spécifier l'utilisateur après la ligne, et ceci s'appliquera donc à tous les utilisateurs:
DefaultRoot ~ signifie que tous les utilisateurs se connecteront dans leur propre répertoire.
Ici, on n'a pas besoin que Merry et Pippin puissent accéder tous les deux au même dossier. Si ça avait été le cas, il aurait été judicieux de créer une arborescence à part, à laquelle les deux peuvent accéder.
On laisse donc ici **DefaultRoot ~**
On peut aussi modifier le nom du serveur FTP, ici on écrit "LaPlateforme".

```
# Set off to disable IPv6 support which is annoying on IPv4 only boxes.
UseIPv6 off
# If set on you can experience a longer connection delay in many cases.
<IfModule mod_ident.c>
  IdentLookups off
</IfModule>

ServerName "LaPlateforme"
# Set to inetd only if you would run proftpd by inetd/xinetd/socket.
# Read README.Debian for more information on proper configuration.
ServerType standalone
DeferWelcome off

# Disable MultilineRFC2228 per https://github.com/proftpd/proftpd/issues
# MultilineRFC2228on
DefaultServer on
ShowSymlinks on

TimeoutNoTransfer 600
TimeoutStalled 600
TimeoutIdle 1200

DisplayLogin welcome.msg
DisplayChdir .message true
ListOptions "-l"

DenyFilter \*.*/

# Use this to jail all users in their homes
DefaultRoot ~

# Users require a valid shell listed in /etc/shells to login.
# Use this directive to release that constrain.
# RequireValidShell off

# Port 21 is the standard FTP port.
Port 21
```

La configuration établie, on redémarre le service:
sudo systemctl restart proftpd

Enfin, on vérifie les ports ouverts sur notre serveur avec la commande ss:

ss -lntu (l pour “listening”, c’est-à-dire les ports en “écoute”, comprendre “en attente de signal”; n pour “number”, c’est-à-dire l’id exacte du socket/port; t pour “tcp”, donc les ports tcp; et u pour “udp”)

```
sysadm@Debian12NG:~$ ss -lntu
Netid     State      Recv-Q     Send-Q         Local Address:Port          Peer Address:Port     Process
udp       UNCONN     0          0                    0.0.0.0:68               0.0.0.0:*
tcp       LISTEN     0          128                  0.0.0.0:6627             0.0.0.0:*
tcp       LISTEN     0          128                  0.0.0.0:21               0.0.0.0:*
tcp       LISTEN     0          128                     [::]:6627                [::]:*
sysadm@Debian12NG:~$
```

On voit ici que le port 21 est en listening, on peut donc se connecter en FTP.
On crée alors les utilisateurs Merry et Pippin, voici la procédure pour Merry (même chose avec Pippin):
sudo useradd -m (spécifier que Merry aura un répertoire automatiquement créé dans le home) Merry
sudo passwd Merry
kalimac

Notons qu’ici, lorsque l’on tape le mot de passe, on n’a aucun caractère de feedback. On peut modifier ceci (pour des questions de confort, même s’il est plus sécurisé d’avoir un mot de passe complètement invisible lorsqu’on le tape) dans le visudo, en rajoutant pwdfeedback après la ligne “env_reset”. Cette spécificité a cependant fait l’objet d’une faille pendant un temps, sur Linux Mint, alors contentons-nous de l’utiliser pour nos machines personnelles dans un cadre d’apprentissage en confort.

Job 10:

Comme client FTP, nous utilisons le plus classique: Filezilla
C’est un logiciel libre, de licence publique GNU, très efficace, le plus répandu et recommandé.
Je l’installe sur ma machine hôte:
sudo apt install filezilla

```
merkava@mantak:~$ sudo apt install filezilla
[sudo] Mot de passe de merkava :
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
filezilla est déjà la version la plus récente (3.63.0-1+deb12u2).
0 mis à jour, 0 nouvellement installés, 0 à enlever et 6 non mis à jour.
```

Il est ici déjà installé mais on comprend le principe.
Je lance Filezilla, rentre “localhost” en nom d’hôte, Merry en utilisateur et “kalimac” en mot de passe, puis 2121 en port (redirigé, comme pour le ssh).
Je crée un fichier texte sur mon ordinateur hôte: **touch mon_fichier.txt** puis il me suffit de faire un glisser-déposer sur FileZilla dans le répertoire voulu.