

# DDWS

## JOB N°1

```
baccam@baccam:~$ sudo apt install openssh-server
[sudo] Mot de passe de baccam :
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
openssh-server est déjà la version la plus récente (1:9.2p1-2+deb12u1).
openssh-server passé en « installé manuellement ».
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
baccam@baccam:~$
```

Après avoir installé ma machine virtuelle Debian, la première étape a consisté à installer le serveur **OpenSSH**.

Pour cela, j'ai ouvert un terminal sur ma machine virtuelle et saisi la commande suivante : **sudo apt-get install openssh-server**. Cette étape est cruciale car elle permet d'activer la possibilité de se connecter à distance via **SSH**.

```
baccam@baccam:~$ sudo systemctl start ssh
```

```
baccam@baccam:~$ sudo systemctl enable ssh
```

Une fois OpenSSH installé, j'ai procédé au démarrage du service **SSH** en utilisant la commande : **sudo service ssh start**. Cette action a immédiatement activé le serveur **SSH** sur ma machine virtuelle, permettant ainsi les connexions distantes.

Pour que le serveur **SSH** démarre automatiquement à chaque redémarrage de la machine virtuelle, j'ai exécuté la commande : **sudo systemctl enable ssh**. Cette configuration garantit que le service **SSH** est toujours opérationnel, même après un redémarrage.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

Installez la dernière version de PowerShell pour de nouvelles fonctionnalités et améliorations ! https://aka.ms/PSWindows

PS C:\Users\tedba> ssh baccam@192.168.20.136
The authenticity of host '192.168.20.136 (192.168.20.136)' can't be established.
ED25519 key fingerprint is SHA256:PDJjWmQpCKp7lgJ+XPONxyxD9xfBy+xxXIMUNCivXmk.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? Y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.20.136' (ED25519) to the list of known hosts.
baccam@192.168.20.136's password: |
```

Ensuite, j'ai basculé sur mon système Windows pour établir une connexion **SSH** avec ma machine virtuelle Debian.

Pour cela, j'ai utilisé la commande : **ssh baccam@192.168.20.136**. Cette commande m'a permis de me connecter de manière sécurisée à ma machine virtuelle Debian depuis mon système Windows.

## JOB N°2

```
baccam@baccam:~$ sudo apt update
Atteint :1 http://security.debian.org/debian-security bookworm-security InRelease
Atteint :2 http://deb.debian.org/debian bookworm InRelease
Atteint :3 http://deb.debian.org/debian bookworm-updates InRelease
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Tous les paquets sont à jour.
baccam@baccam:~$ sudo apt upgrade
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Calcul de la mise à jour... Fait
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
```

Avant de commencer l'installation d'Apache 2, j'ai commencé par mettre à jour mon système. J'ai utilisé les commandes `sudo apt update` pour mettre à jour la liste des paquets disponibles et `sudo apt upgrade` pour effectuer la mise à jour.

```
baccam@baccam:~$ sudo apt install apache2
baccam@baccam:~$ sudo systemctl start apache2
baccam@baccam:~$ sudo systemctl enable apache2
```

Une fois le système à jour, j'ai installé Apache2 en utilisant la commande `sudo apt install apache2`. Cela a téléchargé et installé le serveur web Apache2 sur ma machine.

Après l'installation, j'ai démarré le service Apache2 avec la commande `sudo systemctl start apache2`. Cela a immédiatement mis en marche le serveur web Apache2, le rendant accessible.

Pour que le service Apache2 se lance automatiquement à chaque démarrage de la machine, j'ai exécuté la commande `sudo systemctl enable apache2`. Cette étape a assuré que le serveur web est toujours disponible, même après un redémarrage du système.


```

baccam@baccam:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabl>
   Active: active (running) since Wed 2023-10-25 08:25:03 CEST; 9min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 818 (apache2)
    Tasks: 55 (limit: 4590)
   Memory: 11.9M
      CPU: 199ms
   CGroup: /system.slice/apache2.service
           └─818 /usr/sbin/apache2 -k start
             └─820 /usr/sbin/apache2 -k start
               └─821 /usr/sbin/apache2 -k start

oct. 25 08:25:02 baccam systemd[1]: Starting apache2.service - The Apache HTTP S>
oct. 25 08:25:02 baccam apachectl[814]: AH00557: apache2: apr_sockaddr_info_get(>
oct. 25 08:25:02 baccam apachectl[814]: AH00558: apache2: Could not reliably det>
oct. 25 08:25:03 baccam systemd[1]: Started apache2.service - The Apache HTTP Se>
lines 1-17/17 (END)

```

Ensuite, pour vérifier le statut du service Apache2, j'ai utilisé la commande **sudo systemctl status apache2**. Cela m'a fourni des informations détaillées sur l'état du service, confirmant ainsi qu'il fonctionnait correctement.



## Apache2 Debian Default Page

**It works!**

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

### Configuration Overview

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```

/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf

```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.

Enfin, j'ai vérifié l'adresse IP de ma machine en utilisant la commande **ip a**. J'ai pris note de cette adresse IP pour ensuite l'entrer dans le navigateur Firefox de ma machine hôte. Cela m'a permis d'accéder au site hébergé sur Apache2, et j'ai ainsi pu vérifier que tout fonctionnait comme prévu.

## JOB N°3



Serveur Web	Avantages	Inconvénients
Apache	Open source et gratuit, grande communauté de support.	Peut être moins performant, configuration complexe.
IIS (Microsoft)	Intégration aisée avec Microsoft, bonne prise en charge des technologies MS.	Principalement pour environnements Windows, moins flexible pour non-Microsoft.
Nginx	Très performant et évolutif, excellente gestion des connexions concurrentes.	Configuration complexe pour les débutants.
NetWare	Historiquement utilisé pour les réseaux Novell, sécurité renforcée.	En déclin, peu utilisé actuellement, moins de support et mises à jour.
Google Web Server	Hautes performances, conçu pour les besoins spécifiques de Google.	Non disponible en dehors de Google, non open source.
Domino (IBM)	Large éventail de services, intégration avec produits IBM.	Principalement pour environnements IBM, complexité accrue par rapport à d'autres.

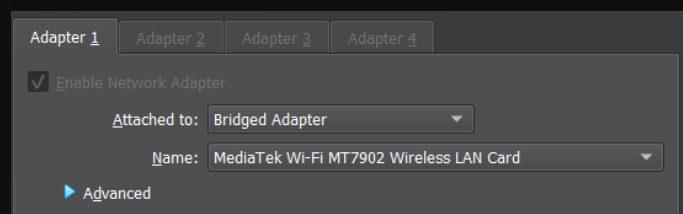
## JOB N°4

```
root@baccam:/# apt install bind9 bind9utils dnsutils
```

```
root@baccam:/home/baccam# apt install ufw
```

```
root@baccam:/# apt install samba
```

Pour commencer, j'ai installé les paquets **bind9**, **bind9utils**, **dnsutils** avec **ufw** et **samba** en avance pour le job n°7 et n°8 car en **bridged**, on risque de ne pas pouvoir le télécharger.



Dans **Virtual Box**, j'ai configuré le réseau en mode "**bridged**," une étape importante pour la suite. (j'ai dû passer de **VMware** à **Virtual Box**, car **VMware** posait des problèmes.)

```
root@baccam:/# hostname -I
10.10.7.177
root@baccam:/#
```

Une fois la configuration réseau en place, j'ai utilisé la commande "**hostname -I**" pour trouver mon adresse IP.

```
root@baccam:/# cd /etc/bind
```

Ensuite, je me suis dirigé vers le répertoire **/etc/bind** en utilisant la commande "**cd**" pour accéder aux fichiers de configuration de Bind.

```
root@baccam:/etc/bind# cp db.local direct
```

J'ai copié le contenu de la zone de base de données (**db.local**) dans un fichier nommé '**direct**'. Cette étape configure le serveur **DNS** en établissant les enregistrements nécessaires pour la résolution des noms de domaine, permettant ainsi de faire correspondre des noms de domaine avec leurs adresses IP.

```
root@baccam:/etc/bind# nano direct
```

```

; BIND data file for local loopback interface
;
$TTL      604800
@         IN      SOA      prepa.com dnsproject.prepa.com. (
                                2          ; Serial
                                604800     ; Refresh
                                86400      ; Retry
                                2419200    ; Expire
                                604800 )   ; Negative Cache TTL
;
@         IN      NS       dnsproject.prepa.com.
dnsproject IN      A        10.10.7.177
www       IN      CNAME    dnsproject.prepa.com.
```

J'ai utilisé l'éditeur de texte 'nano' pour modifier le fichier '**direct**' en y ajoutant mon adresse IP et mon domaine.

Cela permet de personnaliser la configuration **DNS** en associant mon adresse IP à mon domaine, facilitant ainsi la résolution de ce domaine vers l'adresse IP correspondante.

```
root@baccam:/etc/bind# cp direct inverse
```

```
root@baccam:/etc/bind# nano inverse
```

```

; BIND data file for local loopback interface
;
$TTL      604800
@         IN      SOA      prepa.com dnsproject.prepa.com. (
                                2          ; Serial
                                604800     ; Refresh
                                86400      ; Retry
                                2419200    ; Expire
                                604800 )   ; Negative Cache TTL
;
@         IN      NS       dnsproject.prepa.com.
dnsproject IN      A        10.10.7.177
213       IN      PTR      dnsproject.prepa.com.
```

Après avoir modifié le fichier "**direct**", j'ai copié son contenu dans un fichier appelé "**inverse**". La seule modification que j'ai apportée dans ce fichier était à la dernière ligne, où j'ai modifié pour qu'elle ressemble à ceci : "**213 IN PTR dnsproject.prepa.com.**"

Cette modification permet d'associer l'adresse IP 213 à mon domaine "**dnsproject.prepa.com**" pour une résolution inversée.

```
root@baccam:/etc/bind# nano named.conf.local
```

```
//  
// Do any local configuration here  
//  
  
// Consider adding the 1918 zones here, if they are not used in your  
// organization  
//include "/etc/bind/zones.rfc1918";  
zone "prepa.com" IN {  
    type master;  
    file "/etc/bind/direct";  
};  
zone "7.10.10.in-addr-arpa" IN {  
    type master;  
    file "/etc/bind/inverse";  
};
```

Après avoir sauvegardé ces modifications dans le fichier "**inverse**", je suis passé au fichier "**named.conf.local**." Là, j'ai fait des ajustements pour inclure mon adresse IP, mon adresse IP à l'envers (reverse IP), et mon domaine.

Cela permet de configurer le serveur DNS pour qu'il reconnaisse et réponde aux requêtes liées à mon domaine.

```
root@baccam:/etc/bind# nano /etc/resolv.conf
```

```
# Generated by NetworkManager  
search prepa.com  
nameserver 10.10.7.177
```

De plus, j'ai pris en compte le fichier "**resolv.conf**" pour configurer les résolutions DNS sur ma machine en utilisant mon adresse IP et mon nom de domaine "**prepa.com**".

Cela assure que ma machine utilise correctement le serveur DNS que j'ai configuré pour résoudre les noms de domaine.

```
root@baccam:/etc/bind# systemctl restart bind9
```

```
root@baccam:/etc/bind# ping dnsproject.prepa.com  
PING dnsproject.prepa.com (192.168.0.24) 56(84) bytes of data:  
64 bytes from 192.168.0.24 (192.168.0.24): icmp_seq=1 ttl=64 time=0.179 ms  
64 bytes from 192.168.0.24 (192.168.0.24): icmp_seq=2 ttl=64 time=0.049 ms  
64 bytes from 192.168.0.24 (192.168.0.24): icmp_seq=3 ttl=64 time=0.073 ms  
64 bytes from 192.168.0.24 (192.168.0.24): icmp_seq=4 ttl=64 time=0.043 ms
```

Après avoir effectué toutes ces étapes, j'ai redémarré le service **Bind9** et utilisé la commande "**ping**" avec mon domaine "**dnsproject.prepa.com**" pour vérifier que ma configuration DNS fonctionnait correctement.



## JOB N°5

### Comment obtient-on un nom de domaine public ?

Pour obtenir un nom de domaine public, il suffit de choisir un bureau d'enregistrement de noms de domaine, vérifier la disponibilité du nom souhaité, sélectionner l'extension appropriée, l'ajouter au panier, procéder au paiement en fournissant vos informations de contact, configurer les serveurs de noms **DNS**, et ne pas oublier de renouveler le nom de domaine à intervalles réguliers pour le maintenir actif. Ces étapes simples permettent d'obtenir un nom de domaine pour votre site Web ou projet en ligne.

### Quelles sont les spécificités que l'on peut avoir sur certaines extensions de nom de domaine ?

Certaines extensions de nom de domaine ont des caractéristiques particulières. Par exemple, les extensions de pays (**ccTLDs**) sont liées à des régions et peuvent avoir des restrictions géographiques. Les extensions thématiques (**nTLDs**) ciblent des domaines spécifiques, comme .app pour les applications. Les extensions premium peuvent être plus coûteuses en raison de leur attrait. Comprendre ces spécificités est essentiel pour choisir la meilleure extension en fonction de vos besoins.

Voici un tableau avec des exemples :

Type d'Extension	Exemples de Nom de Domaine
Extensions génériques (gTLDs)	.com, .net, .org, .info, exemple.com
Extensions de pays (ccTLDs)	.fr (France), .de (Allemagne), .co.uk (Royaume-Uni), .jp (Japon), exemple.fr
Extensions thématiques (nTLDs)	.app, .io, .blog, .guru, exemple.app
Extensions restreintes	.gov (entités gouvernementales aux États-Unis), .edu (institutions éducatives), .mil (organisations militaires), exemple.gov
Extensions premium	.luxury, .invest, .property, exemple.luxury
Extensions géographiques & culturelles	.paris (pour des entreprises ou sites liés à Paris), .bzh (pour la Bretagne, en France), .nyc (pour New York City), exemple.paris



## JOB N°6

### Modifier les paramètres DNS du réseau

Manuel

**IPv4**

Activé

DNS préféré10.10.0.1

DNS sur HTTPS

Désactivé

Autre DNS10.10.7.177

DNS sur HTTPS

Désactivé

**IPv6**

Désactivé

EnregistrerAnnuler

Si vous avez suivi attentivement les étapes que j'ai fournies du Job n°4, il vous suffit maintenant de mettre à jour les paramètres **DNS** de votre machine hôte.

(10.10.0.1 est le **DNS** de La Plateforme)

Si tout se passe comme prévu, vous devriez être en mesure d'accéder à la page suivante :

A screenshot of a web browser displaying the 'Apache2 Debian Default Page'. The browser's address bar shows 'dnsproject.prepa.com'. The page has a white background with a red header bar containing the Debian logo and the text 'It works!'. Below the header, there is a paragraph of text explaining that the page is the default welcome page for the Apache2 server on Debian systems. It mentions that if the page is visible, the server is working properly, and it provides instructions on where to find the default configuration files. A section titled 'Configuration Overview' follows, explaining that Debian's Apache2 configuration is different from the upstream default and is split into several files. It then lists the files in the /etc/apache2 directory: apache2.conf, ports.conf, mods-enabled, load, conf-enabled, and sites-enabled. At the bottom, there is a bullet point stating that apache2.conf is the main configuration file that includes all remaining configuration files when starting up the web server.

## JOB N°7

```
root@baccam:/home/baccam# ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
root@baccam:/home/baccam#
```

```
root@baccam:/home/baccam# ufw default deny outgoing
Default outgoing policy changed to 'deny'
(be sure to update your rules accordingly)
root@baccam:/home/baccam#
```

Après avoir implémenté le pare-feu **UFW** lors du job 4, j'ai commencé par établir les politiques par défaut pour le pare-feu. J'ai défini la politique de refus entrant en utilisant la commande '**ufw default deny incoming**'.

Cette politique signifie que par défaut, toutes les connexions entrantes sont rejetées, sauf celles pour lesquelles j'ai spécifiquement défini des règles autorisant le trafic entrant.

Ensuite, j'ai configuré la politique de refus sortant en utilisant '**ufw default deny outgoing**'. Cela signifie que par défaut, toutes les connexions sortantes sont également rejetées, à moins que je ne définisse des règles spécifiques permettant le trafic sortant.

```
root@baccam:/home/baccam# ufw allow 80/tcp
```

```
root@baccam:/# sudo ufw allow 139/tcp
```

```
root@baccam:/# sudo ufw allow 445/tcp
```

Pour permettre le trafic entrant sur certains ports, j'ai ajouté des règles spécifiques. Par exemple, j'ai utilisé la commande '**ufw allow 80/tcp**' pour autoriser le trafic **TCP** sur le **port 80**. Cela est couramment utilisé pour permettre le trafic **HTTP** entrant, permettant aux utilisateurs d'accéder à des sites web.

De plus, j'ai autorisé le trafic sur les **ports 139** et **445** en utilisant les commandes '**ufw allow 139/tcp**' et '**ufw allow 445/tcp**'. Ces ports sont associés au protocole **SMB (Server Message Block)** et sont utilisés pour le partage de fichiers et d'imprimantes. En autorisant le trafic sur ces ports, j'ai préparé notre système pour la mise en place ultérieure de services de partage de fichiers **SAMBA**.

```
root@baccam:/home/baccam# nano /etc/ufw/before.rules
```

```
GNU nano 7.2 /etc/ufw/before.rules *
-A ufw-before-input -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A ufw-before-output -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A ufw-before-forward -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT

# drop INVALID packets (logs these in loglevel medium and higher)
-A ufw-before-input -m conntrack --ctstate INVALID -j ufw-logging-deny
-A ufw-before-input -m conntrack --ctstate INVALID -j DROP

# ok icmp codes for INPUT
-A ufw-before-input -p icmp --icmp-type destination-unreachable -j DROP
-A ufw-before-input -p icmp --icmp-type time-exceeded -j DROP
-A ufw-before-input -p icmp --icmp-type parameter-problem -j DROP
-A ufw-before-input -p icmp --icmp-type echo-request -j DROP

# ok icmp code for FORWARD
-A ufw-before-forward -p icmp --icmp-type destination-unreachable -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type time-exceeded -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type parameter-problem -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type echo-request -j ACCEPT
```

Ensuite, j'ai modifié le fichier `before.rules` situé dans `/etc/ufw/` en utilisant la commande `'nano /etc/ufw/before.rules'`. À la ligne `"# ok icmp codes for INPUT"`, j'ai remplacé chaque occurrence de `"ACCEPT"` par `"DROP"`.

Cette modification a pour effet de rejeter par défaut les paquets **ICMP** entrants, renforçant la sécurité en limitant les informations accessibles via ces paquets. Cela contribue à réduire le risque d'attaques basées sur les paquets **ICMP**, améliorant ainsi la sécurité du réseau.

```
root@baccam:/home/baccam# ufw enable
Firewall is active and enabled on system startup
root@baccam:/home/baccam#
```

Après tout ça, il me suffit d'activer le pare-feu **UFW** en utilisant la commande `"ufw enable"`, et tout est en ordre.

## JOB N°8

```
root@baccam:/home# sudo mkdir /home/Partage
```

Maintenant, passons à la mise en place de **Samba**. Pour commencer, j'ai créé un dossier nommé "**Partage**" qui servira bien sûr de dossier de partage entre le serveur et ma machine hôte ou d'autres VM. J'ai utilisé la commande "**sudo mkdir /home/Partage**" pour le mettre en place.

```
root@baccam:/home# sudo nano /etc/samba/smb.conf
```

```
[Partage]
comment = Partage
path = /home/Partage
valid users = @users
force group = users
create mask = 0660
directory mask = 0771
writable = yes
```

Ensuite, j'ai modifié la configuration de **Samba** en accédant au fichier `smb.conf` qui se trouve dans `/etc/samba`. J'ai utilisé la commande "**nano /etc/samba/smb.conf**" pour y accéder et le modifier. Une fois dans le fichier de configuration, tout ce que j'ai fait, c'est d'ajouter le contenu présent dans mon image à la fin du fichier.

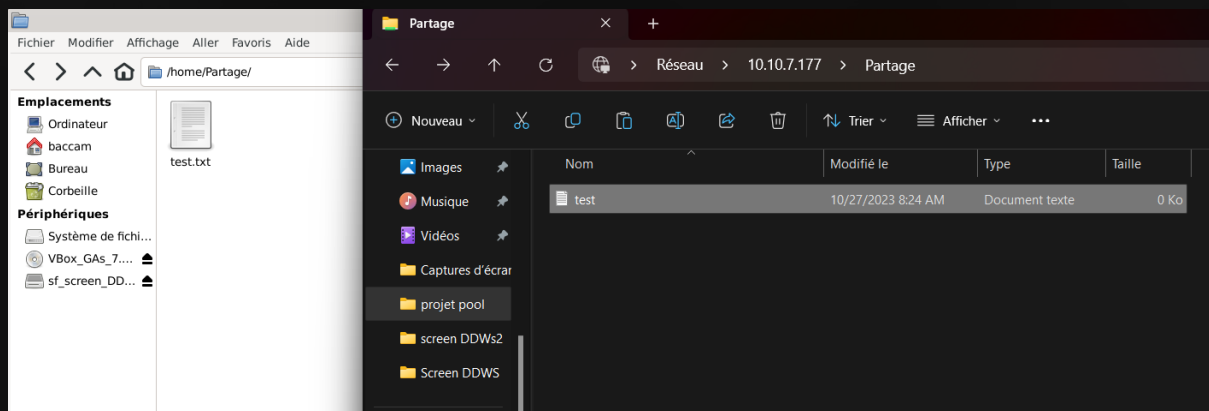
Cela permet de configurer le dossier de partage de manière à ce qu'il soit accessible depuis d'autres machines, facilitant ainsi le partage de fichiers entre le serveur et ma machine hôte ou d'autres VM.

```
root@baccam:/home# sudo smbpasswd -a baccam
New SMB password:
Retype new SMB password:
Added user baccam.
root@baccam:/home#
```

```
root@baccam:/# sudo chmod -R 777 /home/Partage
```

Après cela, tout ce que j'ai eu à faire était de mettre en place un utilisateur et son mot de passe sur SMB pour pouvoir accéder au dossier. J'ai utilisé la commande "**smbpasswd -a [nom utilisateur]**" et j'ai simplement saisi son mot de passe, et tout était en ordre. N'oubliez pas d'accorder les droits appropriés pour pouvoir interagir avec le fichier en utilisant '**chmod -R 777 /home/Partage**'. Cela garantit que l'utilisateur a les permissions nécessaires pour accéder et gérer les fichiers dans le dossier **Partage**.

```
root@baccam:/home# sudo service smb restart
```



Après avoir redémarré le service `smbd` en utilisant la commande '`service smb restart`', tout devrait fonctionner comme illustré dans le screenshot que j'ai suivi pour la configuration.

Le redémarrage du service `smbd` permet aux modifications que nous avons apportées à la configuration de **Samba** d'être prises en compte.

Une fois cela fait, l'utilisateur que nous avons créé précédemment devrait pouvoir accéder au dossier **Partage** avec les autorisations appropriées, permettant ainsi un partage de fichiers fluide et sécurisé.